BlenderArt Magazine Issue 10 | May 2007

blenderart MAGAZINE

Blender learning made easy

Organics

blenderart MAGAZINE

EDITOR/DESIGNER
Gaurav Nawani
gaurav@blenderart.org

MANAGING EDITOR
Sandra Gilbert
sandra@blenderart.org

WEBSITE
Nam Pham
nam@blenderart.org

PROOFER
Kernon Dillon
Phillip A. Ryals
Kevin C. Braun
Derek Marsh
Bruce Westfall

WRITERS
Wong Hua
Juan Pablo Bouza
Javier Galán (chronoh)
Jeremy Ray
David Revoy
Derek Marsh
Erick Ramirez

COVER ART
Enrico Cerica - ' Tomatoes'

Sandra Gilbert
Managing Editor

Welcome to another issue of Blenderart Magazine. In this issue, we take a look at Organic modeling with a nice variety of tutorials and articles covering frogs, humanoid face modeling, a meta-snake and Juan Pablo Bouza shares with us how he created his BlenRig character rigging system.

We also take a look at the ongoing project "Extinction Level Event" and their unique approach to setting up the project using a point based system for credit/shares of the final project. As well as a short walk-through on the creation of their Pyramid Builder Gunboat Poster.

In the Making of David Revoy's "Little Fairy" animation, a wonderfully magical little animation, we learn how he created the amazing castle effect as well as other tips and techniques he used to complete his project.
And since Organic modeling also includes plant life, we have a review of both Arbaro (a stand-alone tree generator) and Ivy Generator (a stand-alone program that allows you to grow ivy in your scene).

So grab a favorite beverage of choice, sit back and enjoy.

Happy Blending!
Sandra@blenderart.org

There are those times when you need trees/plant life in your scene, but either you don't want to model it out yourself or you just don't have the time. Not a problem. We are going to take a look at a couple of stand alone programs that allow you to create some amazing trees and ivy that are easily imported into your scene. Saving you a considerable amount of time that would have been spent if you modeled them yourself.

Arbaro

First up is Arbaro. Arbaro is an easy to use tree generator that allows you to set various tree parameters, then export your tree as a Povray, DXF or Wavefront OBJ file. This makes it compliment nicely with Blender.

So, let's make a simple tree.

1 Open Arbaro (fig 1)
2 There are various parameters you can adjust to customize your tree. For now you can just go with the defaults.
3 Time to export our tree. Push the first tree icon at the top of the screen.
4 An export dialog box will pop up.
5 Choose Wavefront object and choose where to save it.
6 Click start.

Your tree is now ready to import into blender.

7 In Blender, go to File>Import>Wavefront (.obj).

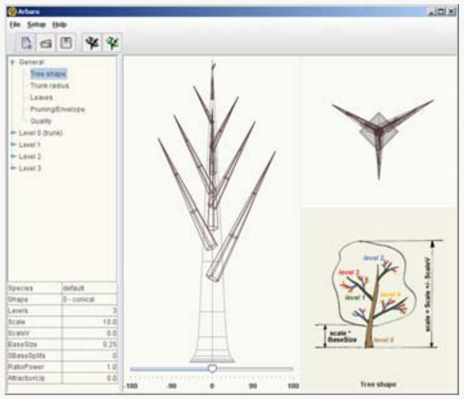


Fig1 Arbaro Main Screen

8 Browse to where you saved your tree and select it.
9 A dialog box will pop up and give you some import options. The default options are good for now.

Your tree is now in Blender, ready for textures and placement in your scene. Arbaro also comes with a variety of preset files for you to quickly create a tree.

10 In Arbaro, go to File>Open.
11 A dialog box will open showing you all the saved preset files.
12 Choose one, export your tree and import into Blender like before.

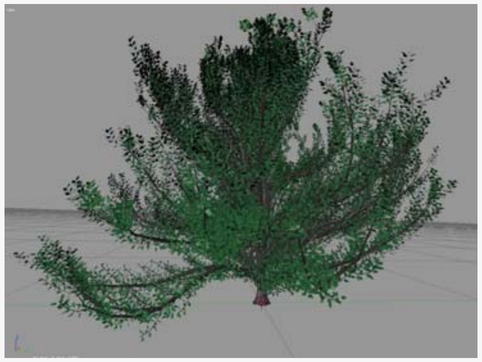


Fig2 Imported tree in blender

Here is a sample of black_tupelo imported in Blender.

You can download Arbaro at: http://arbaro.sourceforge.net/

Ivy Generator

Second up is Ivy Generator. Ivy Generator allows you to grow ivy in your scene. The creator, (Thomas Luft), states that while Ivy Generator does not provide a biological simulation of growing ivy, it does provide a simple approach to producing complex and convincing vegetation that adapts to an existing scene.

The Ivy Generator imports and exports obj+mtl files.

Tim Ellis, (sonix), has written a great tutorial covering how to use Ivy Generator with Blender. The tutorial can be found at the Ivy Generator website as well as in the tutorial forum at www.blenderartists.org.

On the right screen shot show three step ivy generation inside the Ivy Generator.

Ivy Generator, tutorials and Ivy Textures can be found.
http://graphics.uni-konstanz.de/~luft/ivy_generator/

Tim Ellis's, (sonix), Ivy to Blender tutorial can also be found here.
http://blenderartists.org/forum/showthread.php?t=93125&highlight=ivy+generator ■

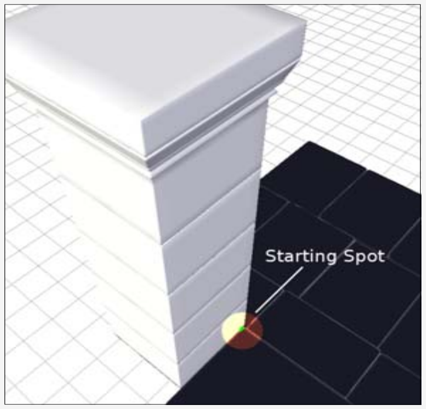


Fig1 Starting spot placement.

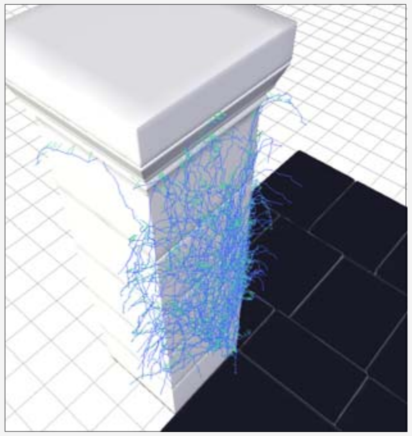


Fig2 Generating the Ivy branches.

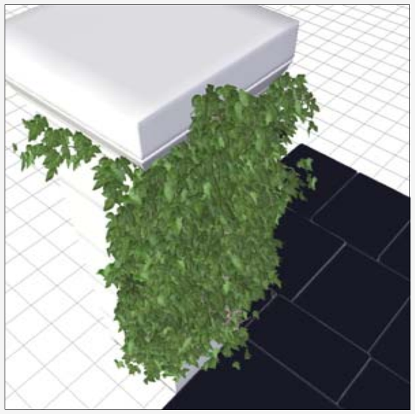
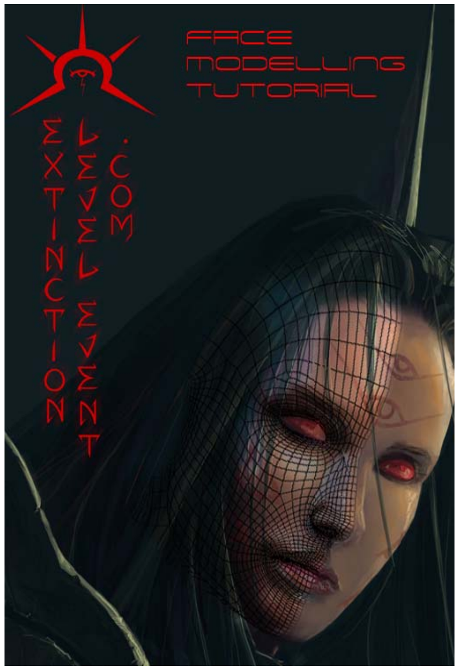


Fig3 Generating leaves.

Taonui Face Modeling

- by Wong Hua



Introduction

There are a lot of methods for modeling a head. Some people like box-modeling, others prefer the more tedious approach of modeling the head polygon-to-polygon.

While at first, box-modeling appears easier, the problems appear when you start detailing the model. Adding loops to the already existing box mesh can be tricky and time consuming for less experienced modelers. Most of them end up with a face that is too dense in poly count, yet less detailed and/or having strange creases and various issues. A lot of the best modelers I've known so far prefer the polygon-to-polygon approach.

Although it is a tedious method, it offers greater control of the final mesh topology than the box-modeling method.

Some very useful tools: Loop Subdivide, and Proportional Editing. Proportional Editing is kind of emulating sculpt but with lower res meshes ... a must have for organic modeling. To use it, go into Edit mode, press the O-key and Mousewheel up or down to change the radius.

Having good references are really a plus. For this face (based on no photography), I used the Marquardt Beauty Mask to have a lot of attractive proportions.

You can find it here : http://www.beautyanalysis.com/mba_youandthemask_page.htm

Take the Frontal Repose and the Lateral Repose for a neutral face. But enough talking! Let's assume you have reached a point where you know the proportions of a human face and wish to solidify this knowledge into a realistic portrait.

I always start by doing the eyes. Eyes are the most recognizable and important part of a face. Some might start with another part of the face, but for me, if I screw up the eyes, I don't even bother finishing the face anyway.

Edge extrusion will be your work horse for new face creation, along with merging vertexes and loop subdivision. Loop subdivision is a nice tool to have, because we essentially work on face loops.

Modeling The Eye

Step1 Add a Plane to the 3D view. Before doing anything, add Mirror then SubD modifiers to the mesh.

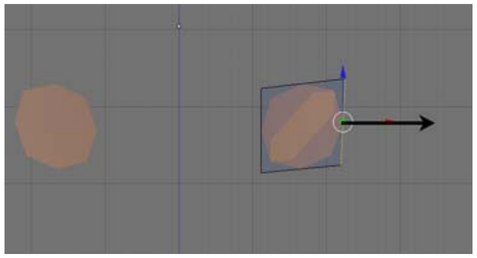


Fig 0. Starting out with a polygon.

Start by extruding an edge from your Plane and do a rough outline of an eye (Fig 1).

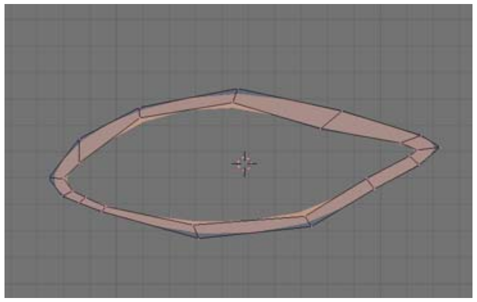


Fig1. Modeling the outline.

Step2 This will serve as a basis for modeling the eyes. Select one edge from the outer rim while pressing [Alt+Shift+D] and extrude the selection, scale it and move the vertices until you have a shape like in fig2.

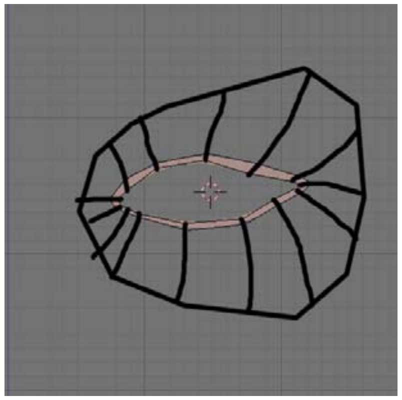


Fig2. Extruding the outer edges.

Step3 select the following edges and subdivide them. I subdivided them 3 times (fig3b).

Step4 Now is a good time to add a Sphere. It will be a perfect placeholder for the eyeballs. For the size, the Sphere's radius is roughly the same than the eye width.

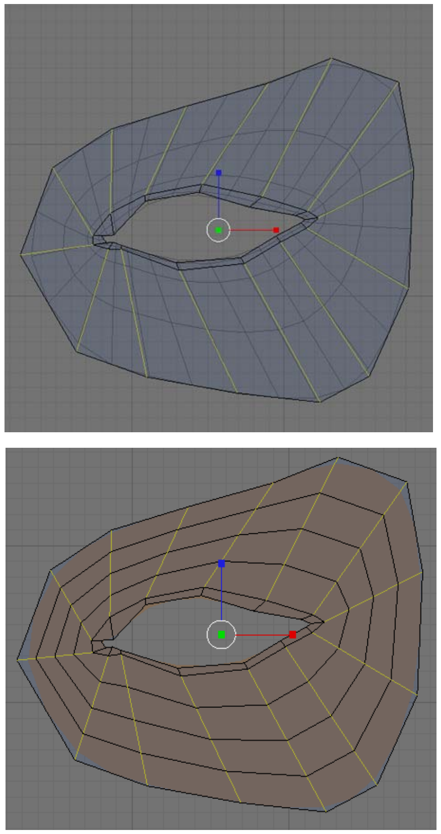


Fig3. Subdividing for detail.

Step5 Now is time to shape the eye. Move the vertices in such a way that the eyelid covers the Sphere. Retopo can be put into good use here.

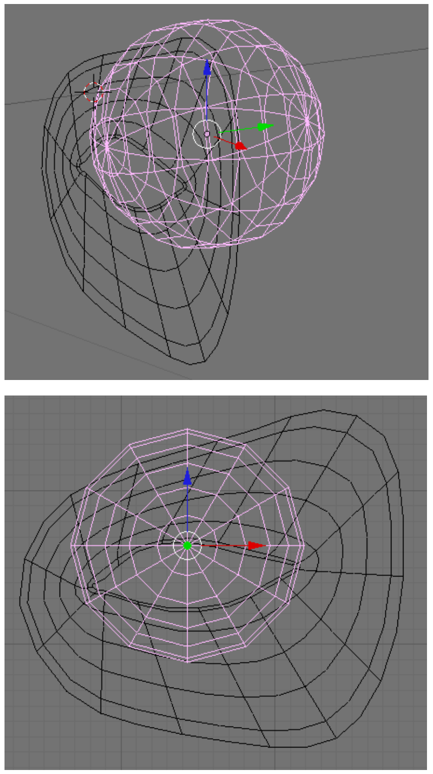


Fig4. Placing an eyeball.

However, note that the vertex moved with Retopo will be too close to the eye's surface. So, you will eventually have to move them again after Retopo.

Now you should have something close to this:

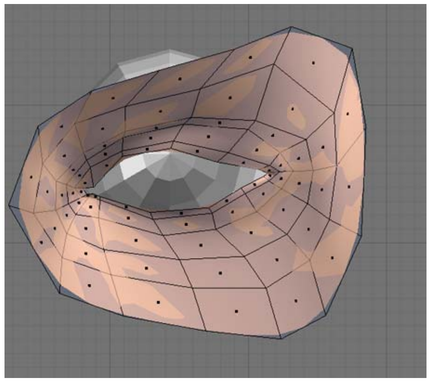


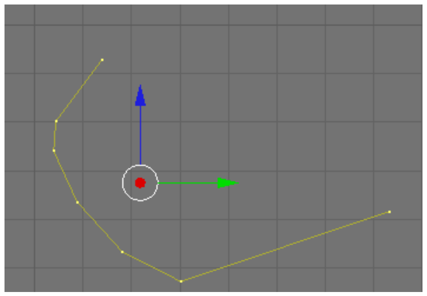
Fig5. Basic mesh for the eye.

Now you have nice looking eyes :) We are done for now. Let's move to the mouth.

Modeling The Mouth

Step1 From the Side view, add a Plane, suppress all but one vertex, select this vertex and start extruding a profile like the one on eye modeling step1.

Once you have your profile, select the edges and extrude them to obtain something close to fig6b.



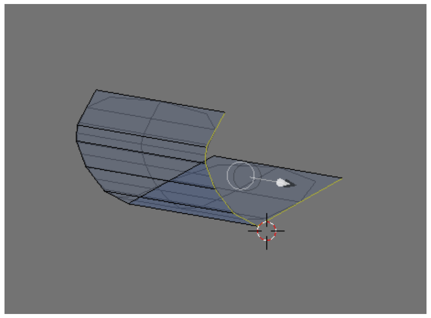


Fig6. Obtaining basic lip shape.

Now that we have these basic lips, let's put them in good shape. Loop subdivide the lower lips and start tweaking the shape as in fig7.

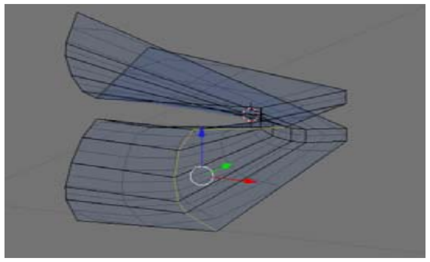
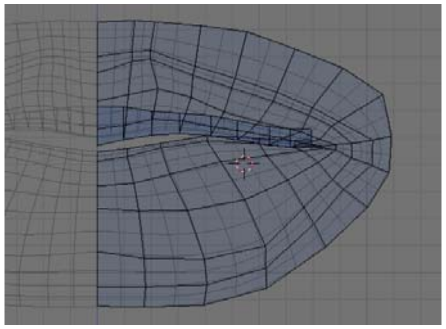
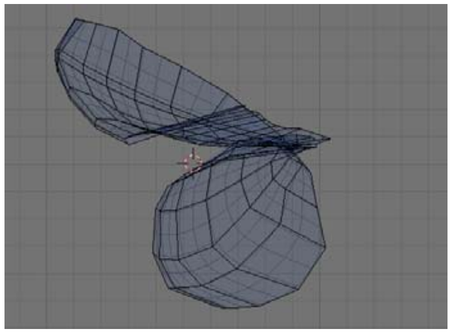


Fig7. Tweaking the lip.

Continue subdividing and tweaking to obtain a shape like in the fig below.





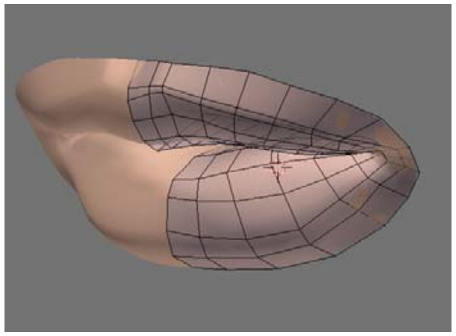
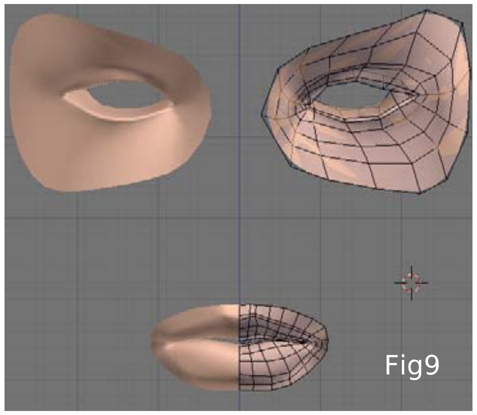


Fig8. Lip shape progress.

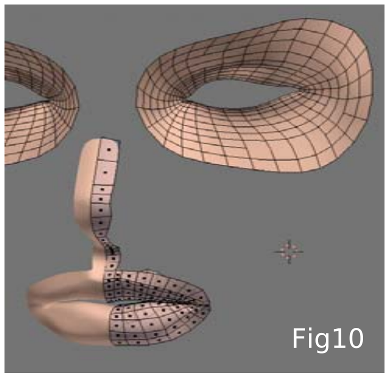
Okay. Now we have decent lips, let's get to the nose!

Right now, this is what we should have:

Modeling The Nose

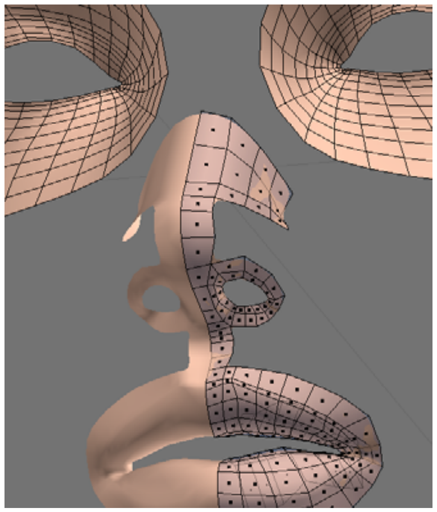


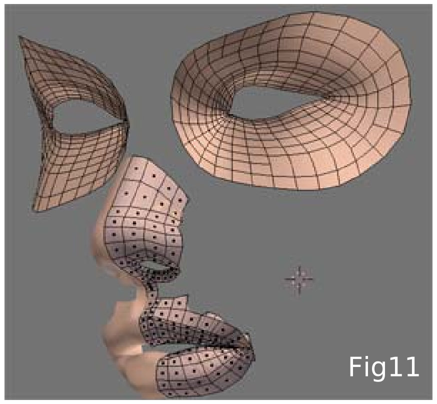
Step1 Who knows where the nose is? Start extruding the nose from the upper lips, like this:



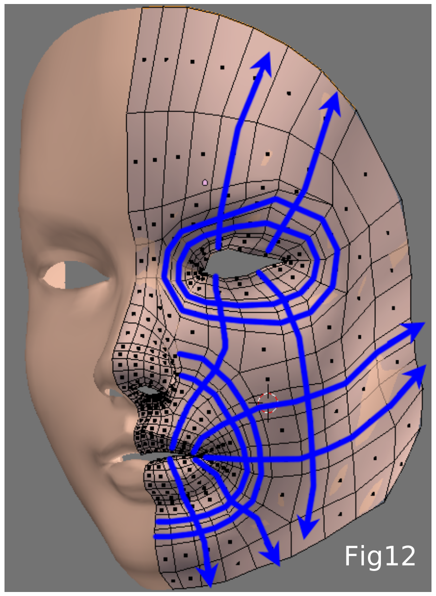
Images below show the nose extruded from one edge of the upper lips.

Use edge extrusion to create prominent nose features (see the image next).



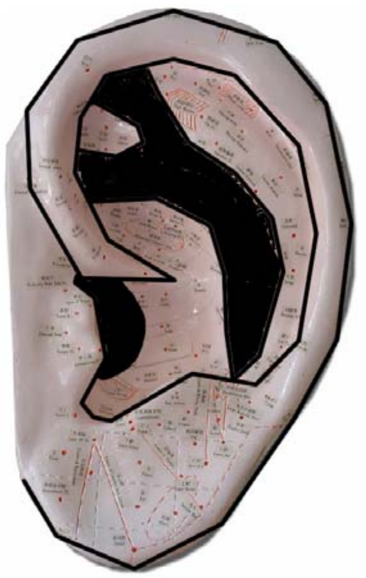


Now extend the facial features by following these guides as shown in fig12.



Modeling The Ears

Step1 People often have trouble with ears. This tutorial doesn't pretend to be right, but to show obvious ear structure that you have to include in order to make your ear feel real. Once the mains structures have been outlined, you see the ear is pretty basic. One big lobe and some kind of a Y shaped bit of cartilage. See the fig13 for reference.



Ear structures are underlined here and you can compare them with with ear reference in fig13a.

Step2 Extrude the lobe you shaped in step 1 and start building the Y shaped bit, see the fig14.

Step3 Here is the next step. Start poly filling the ear. Note the Y shaped thing, and the earhole (yes, the place where you fit you iPod earbuds)

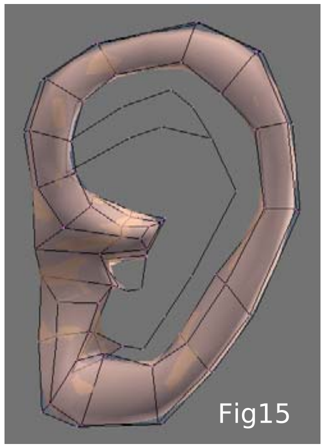
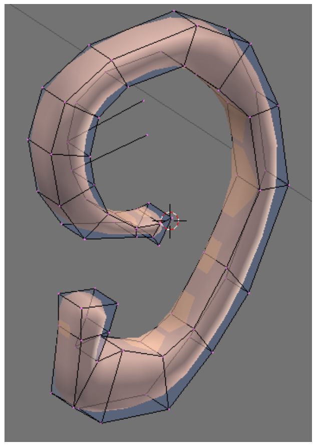


Fig15

Step4 After a while you should have something like this:



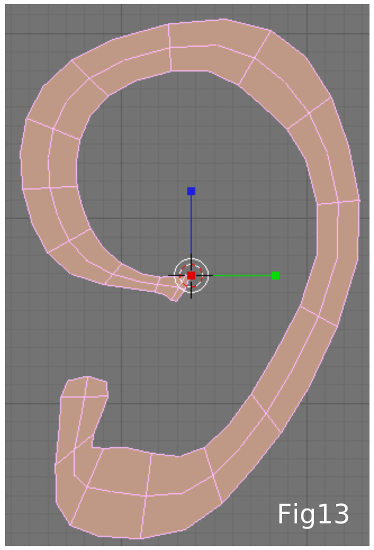


Fig13

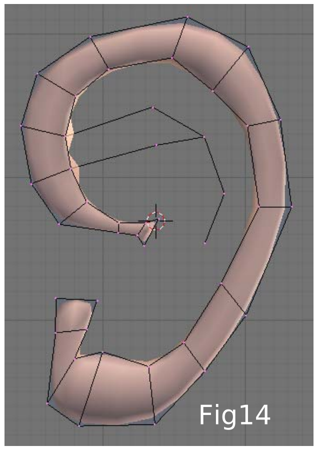


Fig14

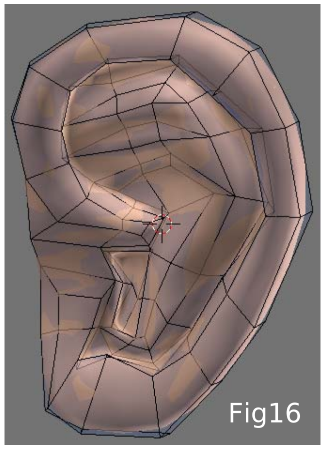
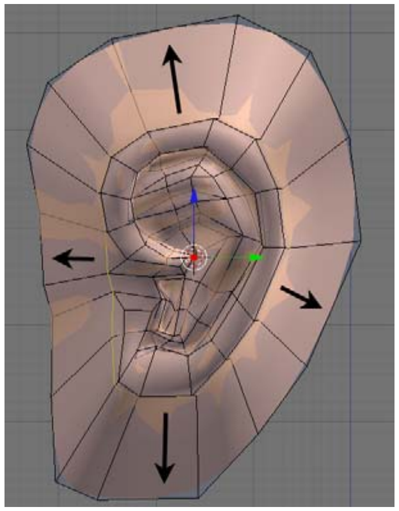


Fig16

Step5 Finish by extruding the outer edge of the ear. Subdivide. From there we will work up to the head.



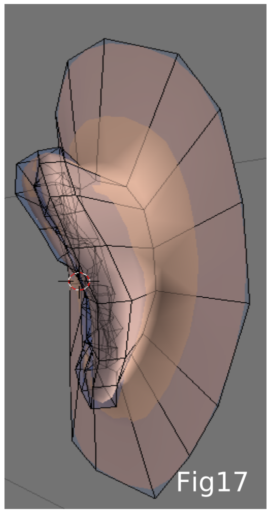


Fig17

Finishing the ear. When you are done, radiate from the ear to model the rest of the head (face not included)

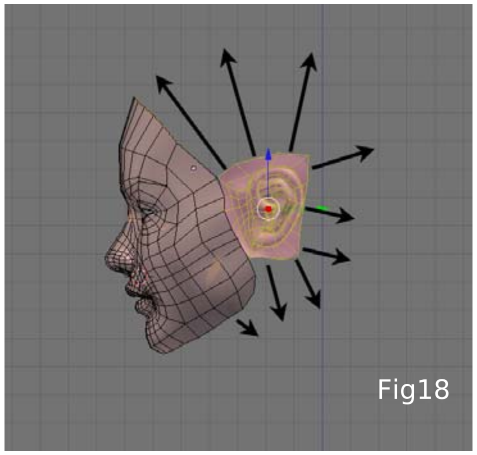


Fig18

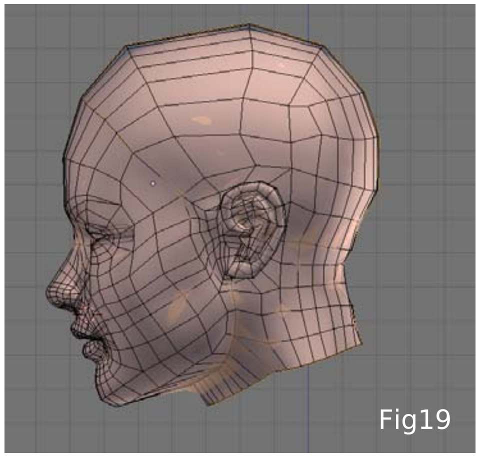


Fig19

.

Finishing the head...

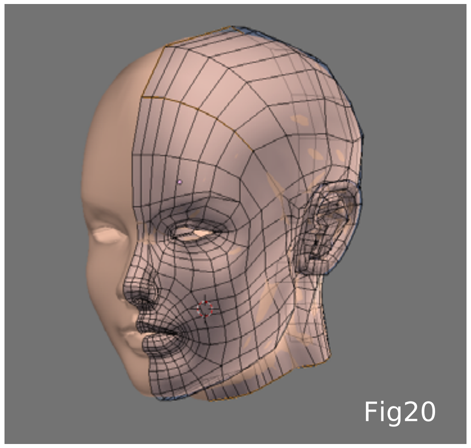


Fig20

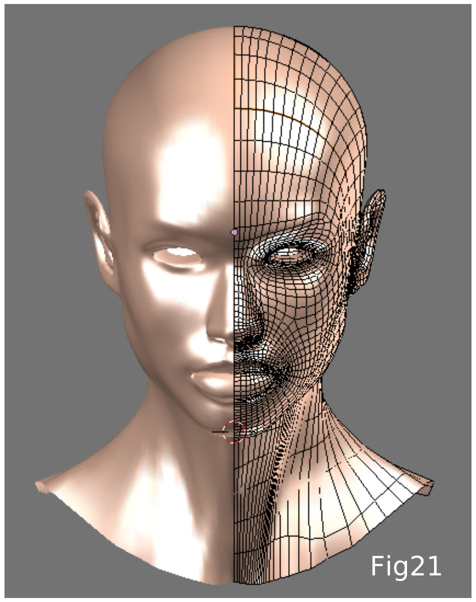
And voilà...

Now you have a face that is presenting desirable features like : relatively low poly count but enough poly density where it is needed.

If you need to build a higher resolution mesh, this face is easy to modify and work with.

For the specific case of the Extinction Level Event project, a higher resolution face was required in order to add finer facial features (*dixit* Jeremy Ray)

No sweat. Using multires and tweaking with sculpt tool, this is what you can do.



Notice that this mesh has twice the poly count compared to the previous head. But, due to its correct topology, (i.e. - the right polys at the right places), reworking the mesh with the Sculpt tool is a walk in the park.

Thanks for reading.

Blender News

New Blender manual "The Essential Blender" goes to print

The new Blender manual, ‘The Essential Blender’, is now at the printers! Expect shipping to begin no later than June 12th.

'Essential Blender', the next book from the Blender Foundation, is your official guide to learning the fundamentals of this open and free 3D software suite. The book can be used as a step-by-step guide for people new to Blender or new to the latest changes in Blender. This book is the ideal companion for the previous 2.3 Blender guide.

'Essential Blender' will get you working with modeling, materials and texturing, lighting, particle systems, several kinds of animation, and rendering. In addition, there are chapters on the new mesh sculpting tools and the brilliant compositor. For users familiar with other 3D packages, there are separate indices that reference topics using your application's terminology. If you've been looking for a way to give yourself a solid foundation in the basic tools and techniques of Blender, 'Essential Blender' is there for you!

Order your copy today, sale prices end June 5th.

Blender 2.44

Blender 2.44 was intended to be an upgrade release, mainly to clear up bugs and ensure a clean stable version before the big push to 2.5, but the developers couldn't resist putting in at least a few interesting new features for us to play with.

This version now is fully 64-bits compatible, new Cast and Smooth modifiers were added, a couple of Composite Nodes were added, and a revamp of the old mesh primitives was done adding new parameters and options. But the most relevant addition to Blender 2.44 is the new long-awaited, but unexpected feature: Subsurface Scattering (SSS) support!

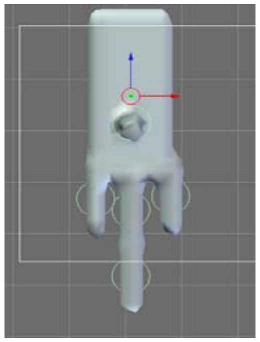
Grab your free copy today.

Managing Metaballs

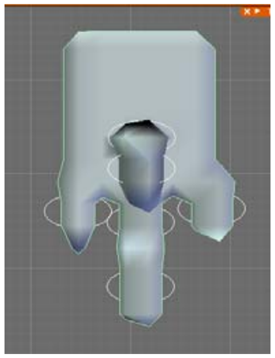
- by Erick Ramirez

Introduction

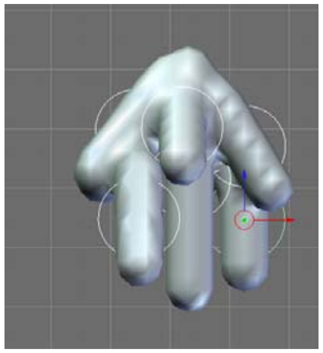
The pros of meta are they work very nice with snap, to make a single mesh, they respond with proportional edit, depending what you are working on, they can quickly build models, among a few other things.



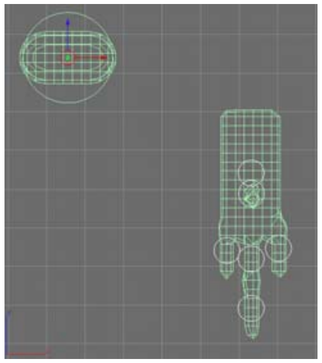
A pattern created in one single object. The meta object handle is selected (individual handles are highlighted, a pattern is created in one single object. The meta object handle is selected (individual handles are highlighted together), it means meta sub-objects can only change their pattern through edit mode. In object mode you can scale this pattern but it doesn't scale similar to the way you would expect when scaling a mesh object. Meta object deformation is around their original axes. The handles deform as you scale, if you scale non-uniform, the handles will deform non-uniform and it will affect the influence of meta sub-objects. You cannot separate Meta sub-objects but you can duplicate the original sub-objects and delete the ones you don't want. For better management you must link this meta object selecting its handle and then selecting any helper object that isn't a meta object, and press [Control+P key] to parent.

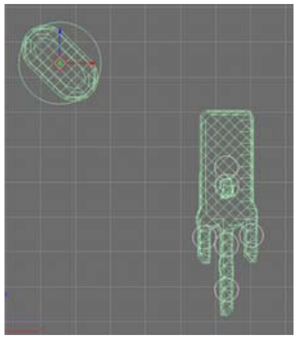


A scaled non-uniform meta pattern. Note the influence of handles is stronger along the horizontal and weaker along the vertical. Meta object selection is different, if you are selecting the meta object directly, you are selecting the entire meta clan. If you select the handle of any meta object except the first one, you are selecting the sub-selection of the meta object. Yet if the meta object selected is a meta pattern you can select the meta-objects in edit mode.



A pattern created by a group of meta objects can change through Object Mode and Edit mode. The advantage is better than a pattern created by sub-objects, even the objects can be sub-objects patterns. The handles can move to the twenty layers known and do a better performance. But the meta objects should be linked to one that acts as a meta leader. At the same time you should add a helper object to this group and link the meta leader to the helper that isn't a meta object for better manageme nt.





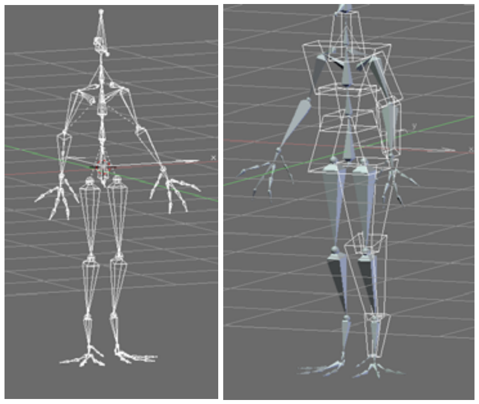
If you select the first meta object through its handle, the whole meta clan is highlighted indicating they had a life link with this one, and if you delete it, the whole meta clan will disappear.

Rotating the first meta object will change the structure for the remaining meta objects even if you scale or move this. This is an important feature, you must protect the life of this one and separate it from the other metas.

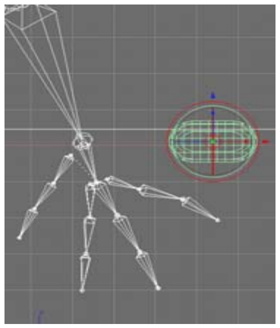
Making A Mudman

This tutorial demonstrates how I've learned to manage metaballs and how this can help to create very complex patterns used for some exigent details.....But metaballs are extremely difficult to manage and require a lot of time. So metaballs features need a lot of practice and training in order to get wonderful features. A powerful PC is required to create with metaballs because the program needs to calculate the coordinates for the crescent number of meta objects, so the more meta object you have, the more your PC starts to slow down.

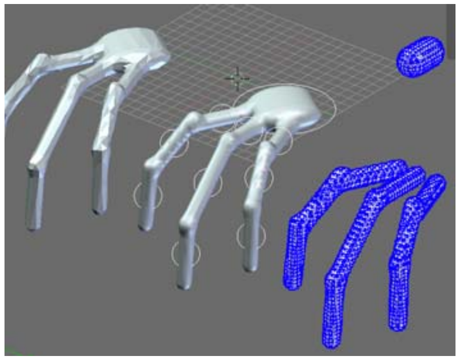
Unfortunately management of meta objects leaves a lot to be desired, but, I have made an effort to create effective managing methods. I started with a armature (I took and edited it from the frog armature). .



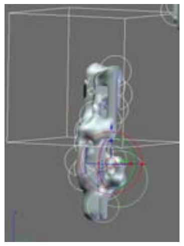
I created cubed geometry for boundaries.The idea is to create an influence reference about features of cross sections (wider, thinner, etc.). Newer geometry is supposed to fit into this boundary.



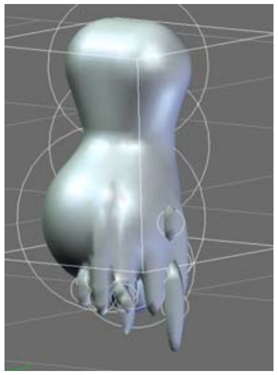
With enough settings I added a meta object [Add>>Meta Tube] for creating the first meta object. To start with, I wanted to create geometry that fit enough for fingers



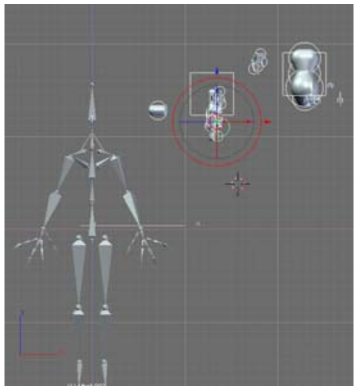
Meshes created by meta objects.



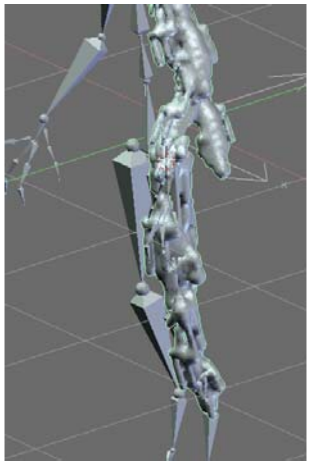
I created a mud pattern with a group, then I linked this group to a box mesh. I created several copies of this by selecting the box and [Spacebar] Select>>Grouped>>Immediate Children.



The body

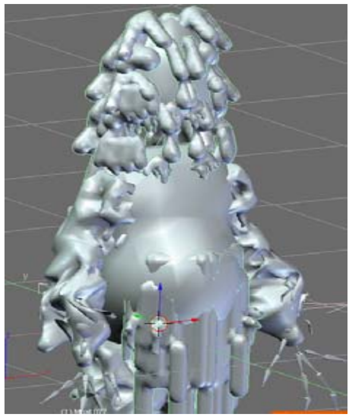


The set



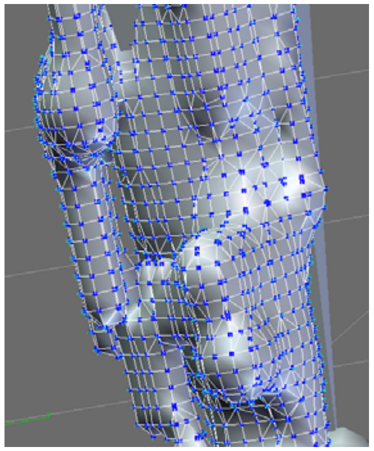
Wrapping the armature. I had several problems wrapping, it seemed the mud would never fit the space.

Result after several experiments.

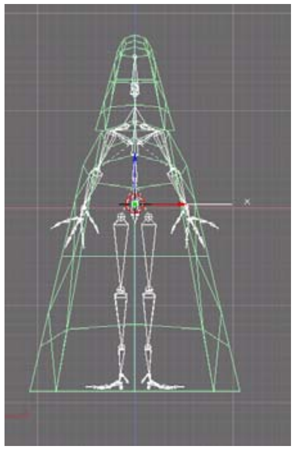


The conceptual design for my mudman.

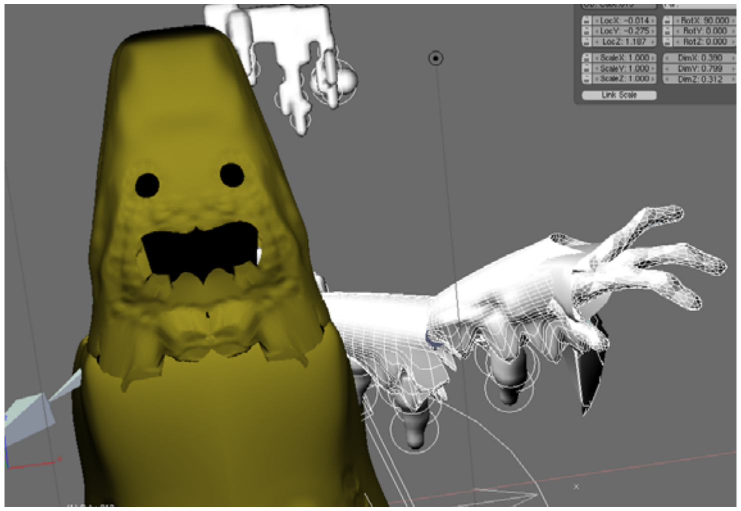
The result wasn't exactly what I wanted, so the metaballs must be managed with better methods.



Face flow resulting by converting meta-objects into mesh. The decimate modifier for reducing vertex quantity didn't work in this case , so I had to use python scripts (to use them you must have python 2.4 installed for newer Blender's versions). Select the Poly Reducer while in Edit mode, Mesh Menu (Header)>>Scripts>>Poly Reducer and a pop up will appear.



Well, redesigning the concept a little like Darth Vader.



Not finished yet, here is my mudman. In this case I applied the multires mode for the mesh, then Sculpt mode for creating a mud texture and meta-objects for details.

Conclusion

The main feature and advantage that meta object give us is creating a fast union, or a subtraction without much effort, not exactly for creating large geometry.
Perhaps it is a lot more helpful in creating superficial patterns, and some very complex patterns

Modeling And Rigging A Frog

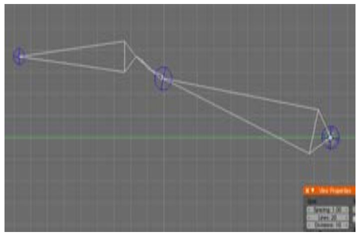
- by Erick Ramirez

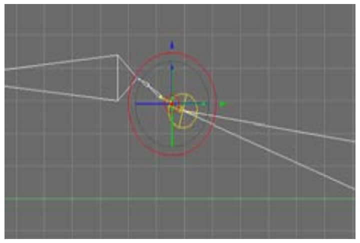
Introduction

This tutorial will show you how to construct geometry reference knots for building an animal like a frog, and show the problems that I had to face.

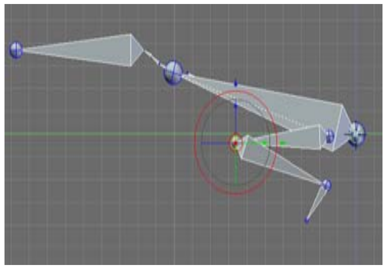
Setting Knots (Rig Creation)

In front view Add>>Armature to create the first link. Go to right view and modify as above and extrude 4 times to create the neck bones and the head. Try to create similar proportions.

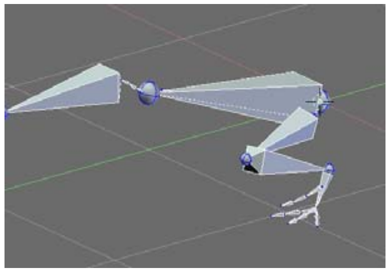




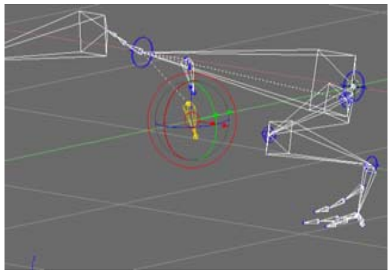
Select the second link and [Shift+D key] to duplicate the link and move it to the right side. In front view move it to the right side to create a leg.



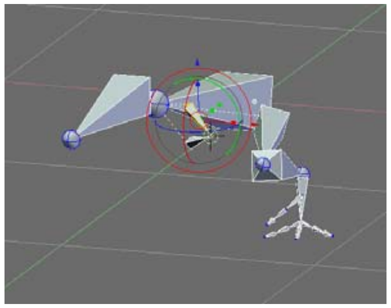
Create the leg limb by extruding this bone as above.



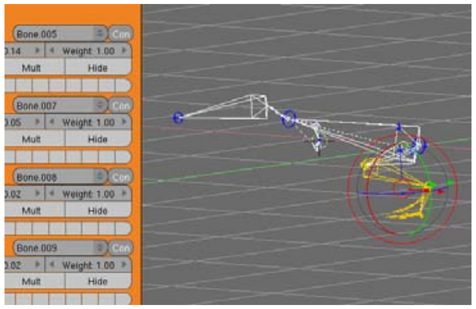
Create the fingers by duplicating finger links.



Create the shoulder blade using the same procedure, but now move over the blade area, duplicate the bone to create a clavicle and relocate in to its respective area. Name them Blade_L and Clavicle_L respectively



'Fuse' the clavicle with the blade. Select the bones shown above and press [Shift+D key] to duplicate the bones. Move them forward.



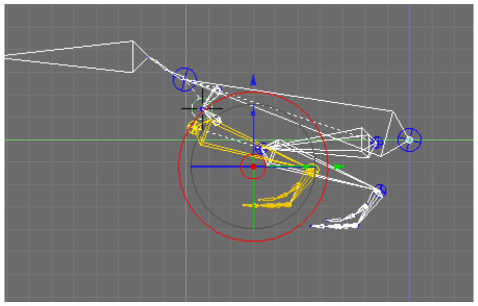
Bone.005 Con
0.14 Weight 1.00
Mult Hide

Bone.007 Con
0.05 Weight 1.00
Mult Hide

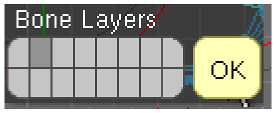
Bone.008 Con
0.02 Weight 1.00
Mult Hide

Bone.009 Con
0.02 Weight 1.00
Mult Hide

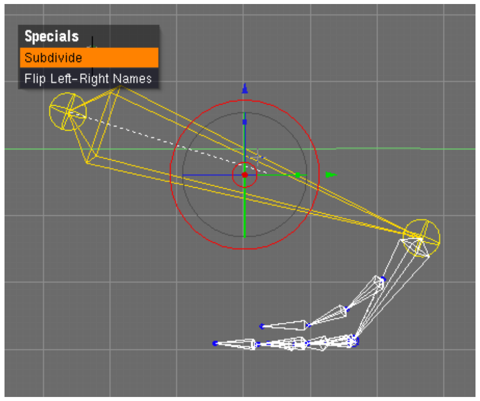
Align the new bones with the clavicle. Select the arm bone and on the Armature Bones Panel change the parent and select Clavicle_L, then press the connect button to connect it to the clavicle.



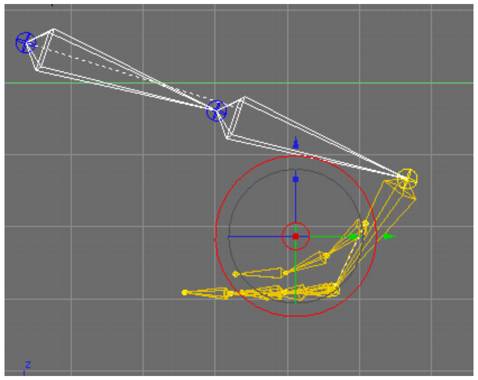
Go to pose mode and press M to move the bones to another bone layer. Back to edit mode [TAB key] and on the Armature Panel [F9] select the next layer where you moved the bones.



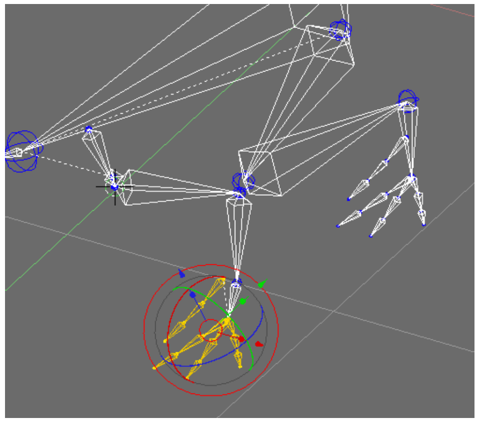
Select this bone and press E key to open the specials dialog box and select subdivide.



Now select the frog hand shown above and move it forward.



Move the fingers a little to adjust them.

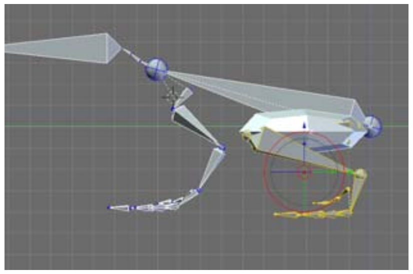


Create a cube and select the subsurf modifier for it and move it to the frog femur. Select pivot 'bounding box center' and move the vertices to get a primitive shape for wrapping the femur bone. Duplicate the cube to wrap the bottom leg.

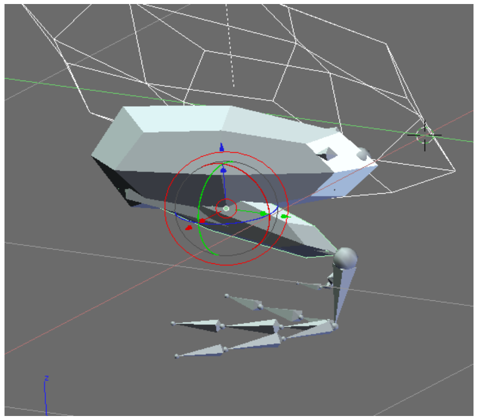
Go to pose mode for the armature and parent the cubes to their respective bones. How?. Select objects in this order: Select the femur box and [Shift] select femur bone (pose mode) and [Control+P key] a dialog box displays and select 'parent to bone'. Then select bottom leg box and [Shift] select bottom leg bone and parent. You can see the result in the Frog.blend.

Reverse Engineering

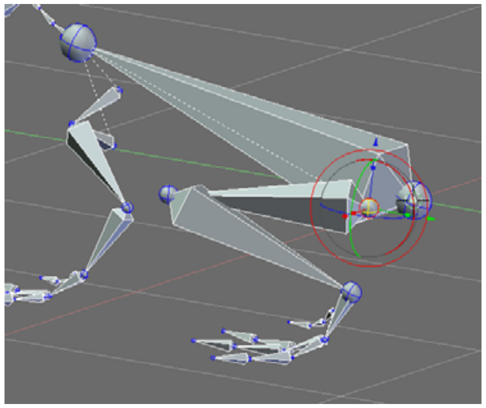
At this moment the knots seem fine, but a frog's living style is frenetic because in evolution the frogs are intermediate between water and ground, then they're excessively dynamic. Hunters and hunted. Their means of defence are their rear limbs as springs always ready to act in danger situations. As the figure shows above, the back legs don't have enough of a power spring form, the pelvis is quite high, so we must adjust this feature.



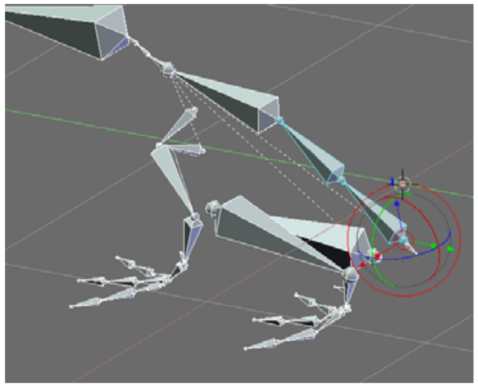
Note that as you move the bone pivot positions the boxes adapt to new forms.



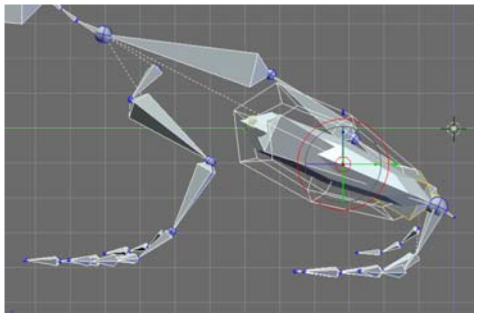
Move the bones low enough that the pelvis almost touches the ground for a full crouch.



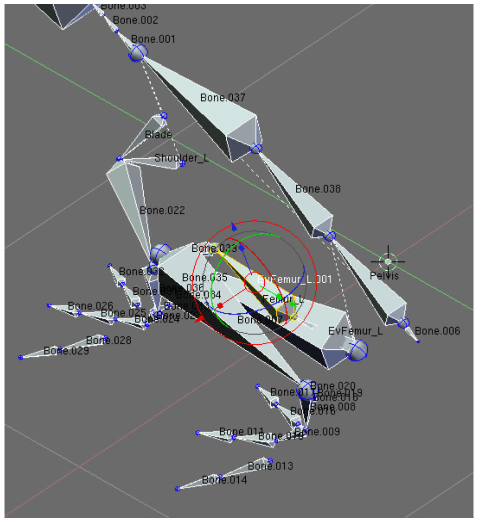
Subdivide the bone shown and adjust it to get a curved spine. This spine actually acts like spring too, similar to cats when they want to jump high.



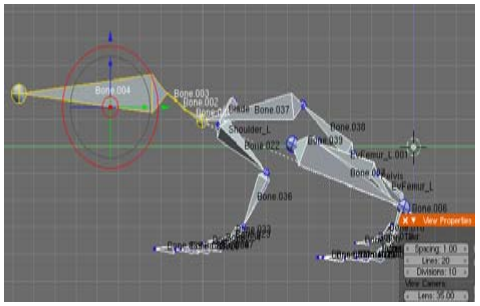
The knees perhaps have to be a little high like hoping insects, but these legs are not only for jumping. The form of the boxes doesn't matter now.



The legs have a little problem because they overlap themselves, the Octahedron form helps to see this problem acting as primitive form. The femur and bottom leg are too tight. In top view try to move the knee over a little. I tried to resolve this problem by creating the bone curvature for the femur and bottom leg. The resulting new bones helped me to improve the box forms.

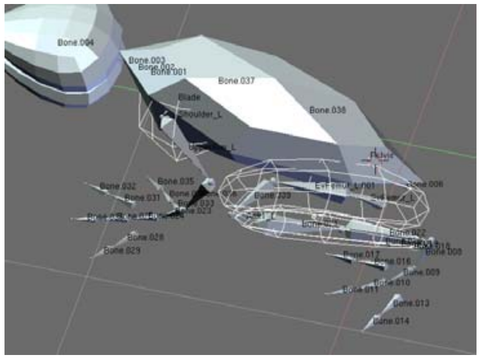


The head seems to be a little big, so let's adjust that.



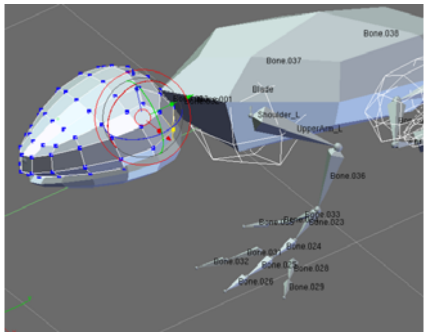
Modeling the Frog

Continue the last session or open



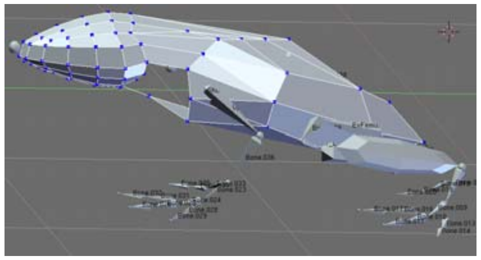
Frog1.blend

Create boxes for other parts of the frog. In top view create a cube and rotate it 45 degrees and adjust the vertices to create the body, add a subsurf modifier to it. Remember to set 'pivot bounding box center' to scale faces and manipulate vertex groups....To get the head, create a cube with a subsurf modifier with



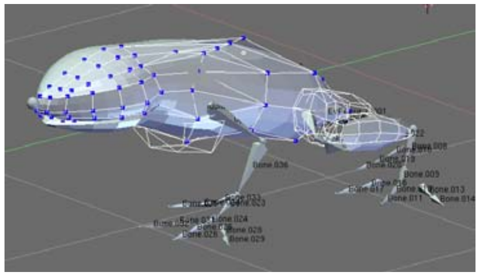
levels set to 2 to get the head shape shown above.

When the head form is OK, apply the subsurf modifier to get this form. Apply the subsurf modifier for the



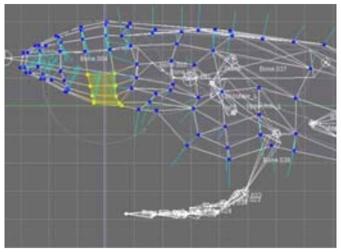
body when you are done. You can see the result in Frog2.blend.

Join both meshes, delete some

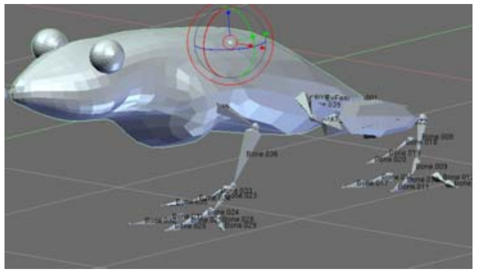


vertices, and then create a bridge to join them into one single mesh.

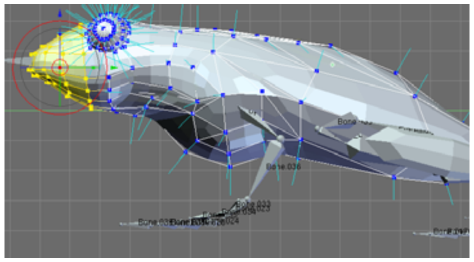
Adjust as needed and attach other shapes like a box for the chest,



adjust the head if it is too big. You can see the result in Frog3.blend.

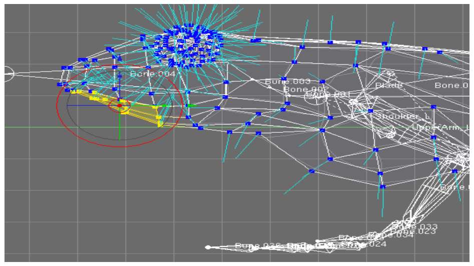


Adjust for frog's head as needed and for smoother results try proportional edit.

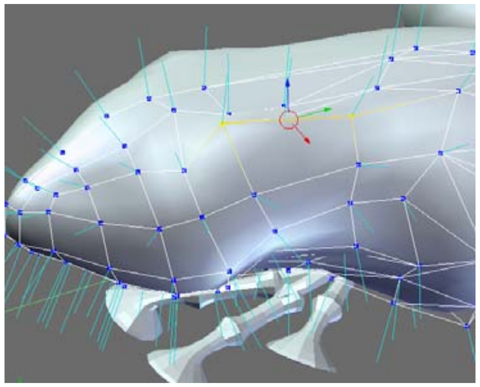


Add a UV sphere to main mesh, then add a subsurf modifier.

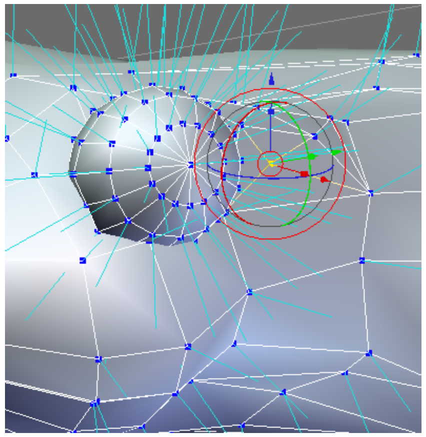
Add some adjustments to the head, that is about half body length.



Add a mouth splitting the mesh and merging some vertices.

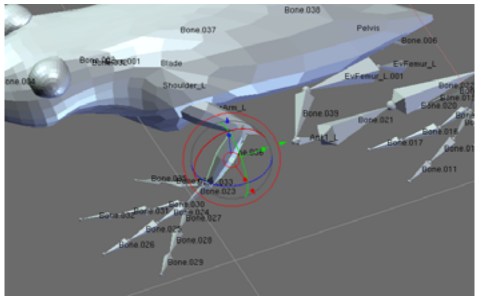


Hide the eye sub mesh (H key), and make some adjustments over the ear section (frogs don't have ears, but holes where they can hear).

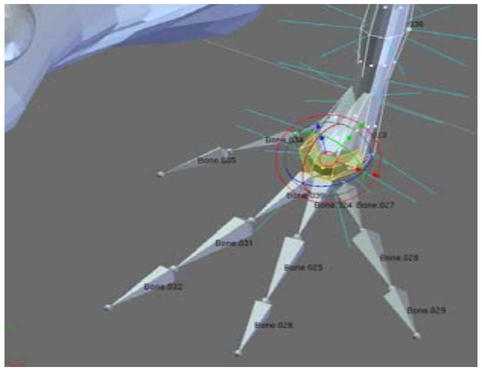


Add some vertices over the ear area to create a depression.

Adding More Features

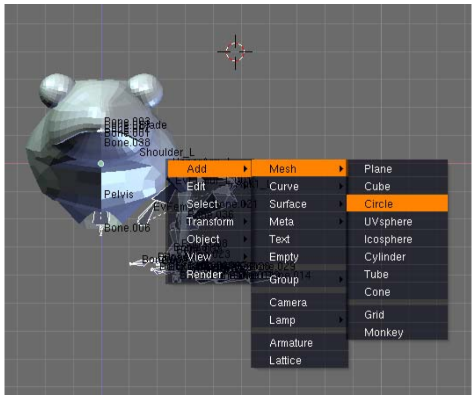


Finish the model by wrapping more subsurfed boxes to the bones that are left.

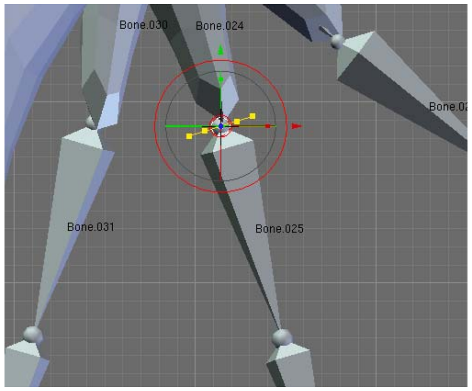


Apply a subsurf modifier to this mesh and adjust it to get 3 faces, Extrude them to form the base of the fingers..

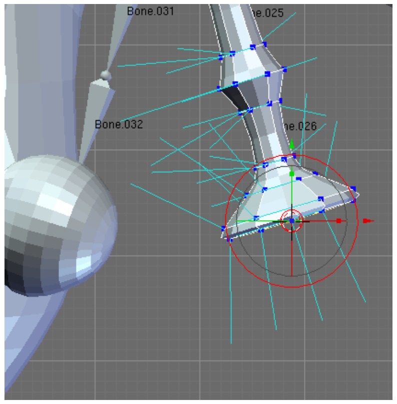
Now to create the fingers, go to front view [1 numeric pad] and [Space Bar] Add>>Mesh>>Circle>>6 vertice circle to get a hexagon shape. Add a



subsurf modifier to it and scale it a bit to get a finger cross section.

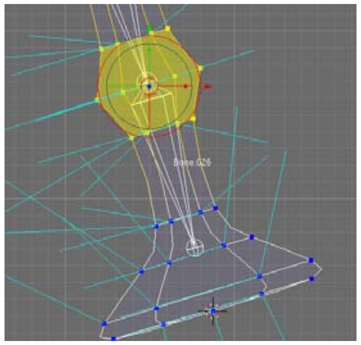


Relocate this shape by positioning the cursor ([Shift+S key] Cursor>>Selection) over the pivot shown above. Select the circle we created and in object mode [Shift+S key] Selection>>Cursor to snap the circle over finger.

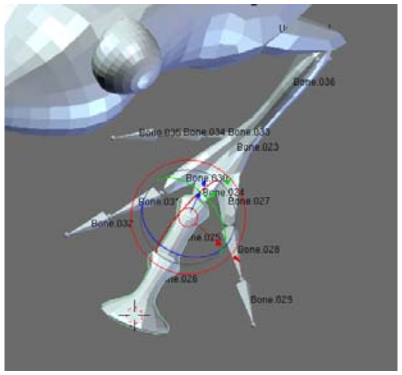


With the PIVOT options set to 'Bounding box center' go to Top view and extrude the hexagon forward to create a frog finger, for the knots extrude, position and scale a little, then you must get back to the original width and finally create a umbrella shape. To close the umbrella, with the latter vertices still selected, [Shift+S] Cursor>>Selection and press E key, then Esc key and W key>> merge>> at cursor or at center or collapse.

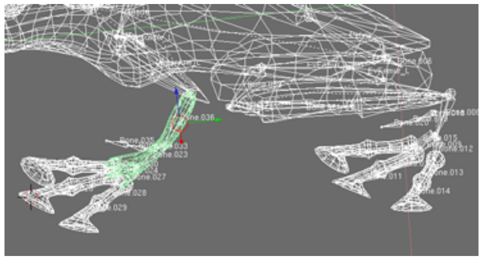
To get the rear knot you can duplicate this knot and merge some vertices.



Make the needed adjustments to get this shape for the finger. The tip may



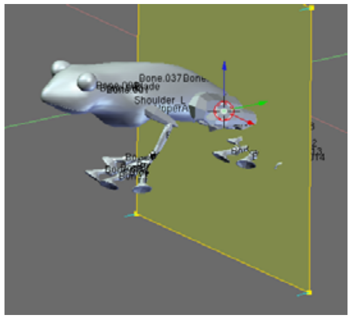
be a suction device to stick on walls or something like that. You can see the result in Frog4.blend.
Make several copies of this finger and adapt them to create others. Make a



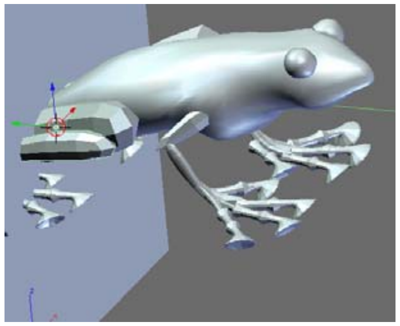
similar mesh for the rear hand. Actually the frogs have different rear hands that serve as flippers, umbrellas, etc, so in this case I preferred to make my frog reflect that fact, perhaps it is a mutant frog that lives in a bathroom hanging over the walls and ceilings like spiderman. Still, attach each finger to the next mesh to get a single mesh.

Mirroring Limbs

Now, in Front view create a plane mesh at the center. Join each box limb one by one with the plane, select any box and select the plane (object mode) and [Control+J key] to join the meshes, select a plane mesh vertex and press L key to select linked vertexes, press P to separate. Next, the next mesh to plane mesh......until complete for all meshes except the body mesh.



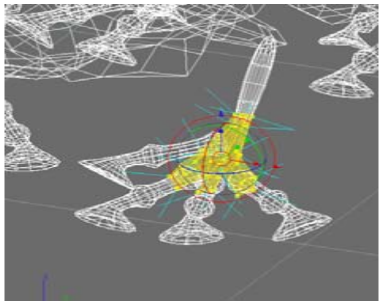
This procedure is for moving the center mesh (actually the mesh pivot in object mode) to the center. You can achieve this by [Space Bar]>>transform>>Center Cursor, and [Control+A key]>>Apply scale and rotation, but in the next step we going to mirror these features and we need assure that all meshes have similar relative UCS (Universal Coordinate System).



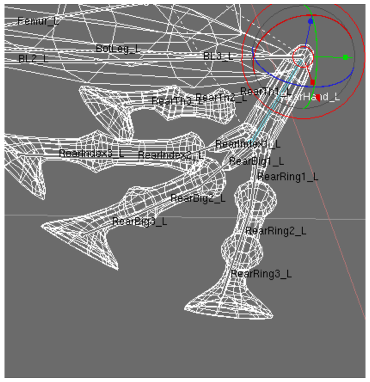
So when you are done, you can add a mirror modifier to each mesh to get the symmetric part. Select the arm mesh and detach some of the palm mesh in the forehand area. Do the same for the rear hand. You can see the result in Frog5.blend and Frog6.blend. In the Frog6.blend I started to give names to the bones.

Advance Editing

Open Frog7.blend. This file contains the entire armature of the frog. If you take a look you can see why the mirrored bone objects have mirrored editing and how this setting can help a lot in avoiding and correcting errors.

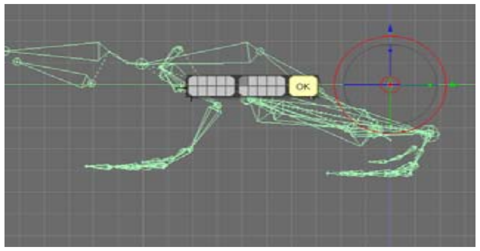


Now, start to apply a mirror modifier to each limb mesh and when the mirrors have been applied, select each mesh again and detach each linked mesh. You can see the result in Frog8.blend.

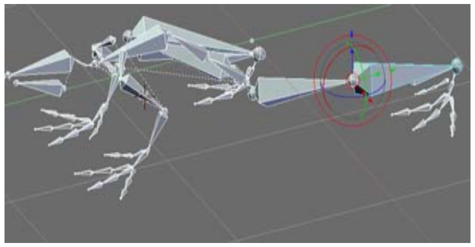


Oops!!. It seems that the rear hand is actually -the ankle!!???....-If you did all the exercises, select rear hand bone and in edit mode W key to subdivide. Now we have an ankle and a rear hand. Finish by parenting the meshes with their respective bones. You can see the result in Frog9.blend.

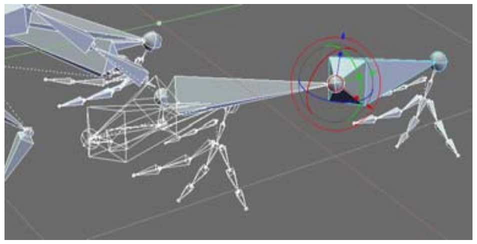
Changing Pose



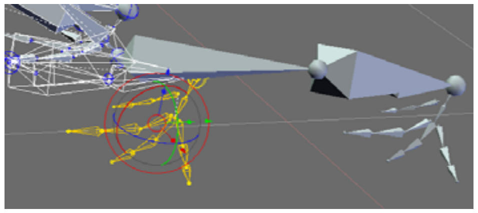
Go to Object Mode for the armature and duplicate it. Move the duplicate to another layer. Select the original armature and on Draw panel [F7] select Wire option.



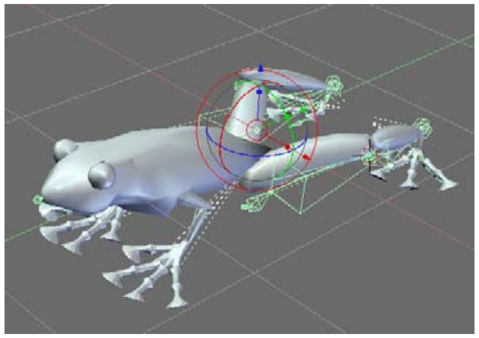
Go and Select the duplicated armature and enter pose mode. Rotate the femur in Top view and rotate the bottom leg to get this pose. Save your file with another number or name.



[Shift] select the layer where original armature is.



Go to edit mode and select the following bones and move them to where the duplicated bones are. Select the duplicate armature and in pose mode select the bottom leg and [Shift+S key]>>Cursor>>Selection to locate the cursor over knee.



Select the original armature and select the knee pivot in edit mode and [Shift+S key]>>Selection Cursor. The legs will occupy the new position, go to Object mode and un-hide the other layers to see the meshes updated.

You can see the result in Frog10.blend.

When you are happy with the actual pose, you can start to join all the meshes as you wish. You can add extra limbs or more heads to get our era touch. Sometimes I need extra arms too. But, being serious, this model has potential to create other living forms and geometry references by evolving into new structures. A weird method perhaps to create simple geometry but, rather easy???

This model is not finished but it has enough information for you to experiment and improve on your own ■

BlenRig An Introduction

- by *Juan Pablo Bouza*

Introduction

I remember being a kid and spending my weekend afternoons watching TV. Oh, I loved it when they transmitted Star Wars documentaries on how they did the stop motion shots, or Jim Henson,s movies, I used to become immersed in that world of puppets, Fantasy, Science Fiction and special effects.

As a kid I had always wanted to be able to get my hands on one of those puppets. But then, one day, the 3D world came into my life.

It took me years to learn the crafts of 3D, and the more experienced I got, the more technical problems arose. Specially when it came to the character animation process. I used to see how the proprietary programs expanded their capabilities and became more and more expensive, and dreamt about buying one of those. But I never had the money to do so.

Then I discovered Blender and most of all, the Blender Community. These guys are really nice people.

So, one day, not so long ago I said to myself, "What the heck, I'm not gonna spend my life waiting for someone to develop a muscle simulation feature for Blender!", I sat tight and started to figure out how to do that with the tools I had at my disposal. So I started **BlenRig**.

There are some who say that this rig has too many bones, I must agree with that. And this fact may frighten you at first glance, but once you get the feel of the rig you will see it is not that hard to start working with it (Manual is available as well). And the results are definitely worth it. Besides, I have never been fond of creating hundreds of shape keys to correct bad mesh deformation.

Well, this is **BlenRig**: an armature capable of preventing a character's skin from shrinking and from having those awful glitches we usually see in 3D. Furthermore, it comes with a complete set of bones for **Facial Animation**.

It's worth mentioning that I designed BlenRig with myself, as an animator, in mind. So, I didn't use fancy controllers or hidden features. There is absolutely nothing scripted or action based. It is all about constraints. I personally like the feel of the model as you animate it, I think of it as a clay model, so, if I want to move the character's cheek, I'll move the bone that is over the cheek and that's it. I tried to keep everything as straight forward and intuitive as I could.

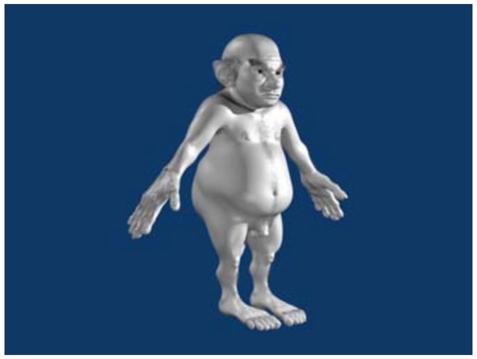
A clay model...that's a good way to describe how I like the animation process to be.

Getting a little bit more technical

As you may already know, one of the big issues of animating a 3D character is that the skin of the model tends to bend inwards at the center of the of the joints, as if it was loosing mass. Another big issue is that you can't easily simulate skin sliding.

With those two topics in mind I started to figure out a bone based mechanism that would let me achieve my goal. At the same time, I did a little bit of research through the web and found a few 3D artists that had gone through the same path I was going. And that encouraged me to carry on with the project.

So, I grabbed a dwarf-like character I was working on, which was actually based on a humanoid mesh I had modeled long ago, and started rigging. Blender's new Sculpt Tools and MultiRes modeling are amazing.

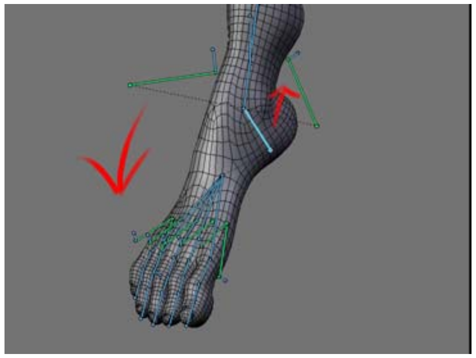


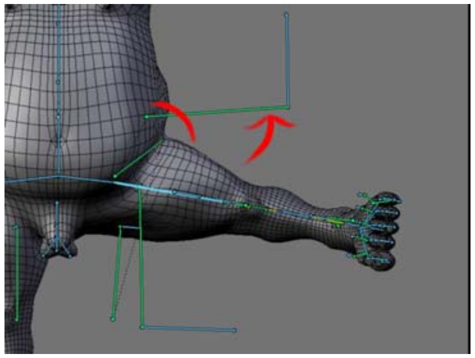
I decided to go for the vertex group technique because I knew I would need to have much more control over the deformation than what the envelopes technique could give me.

Besides, envelopes seem to be much slower than vertex group deformation.

In order to achieve the deformation I was looking for, I knew I had to develop a kind of "Pull up" mechanism.

First I constructed a sort of hammer like armature that would rotate along with the limbs and pull the skin up at the articulation point.





Finally it came, the day I reached for the shoulders ...

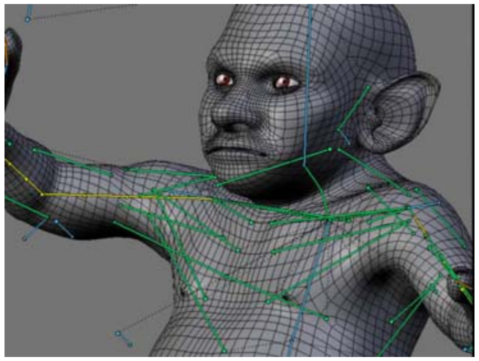
Shoulders, a rigger's nightmare

Everything was perfect, the legs were entirely rigged and beautifully moving, the torso was done and the head was doing a nice job over that little dwarf's neck. But then, I started setting up the bones for the shoulders.

It seemed just impossible to achieve proper deformation, nothing I tried was working, when the arm rotated fine in one direction it didn't on the other one, skin was shrinking, shoulder rotation was disastrous, the chest as well as the back were full of glitches when the arm moved, I was starting to get really desperate.

I spent three entire weeks trying different approaches, and nothing seemed to work. I was about to give up, but my wife encouraged me to carry on and on. Everyday, when I came back home and told her that it was impossible, that I couldn't do it, she would tell me to keep up with the project. -"If you've come this far, you will surely make it with those shoulders" - she said.

Well, then the day came when I could finally achieve good shoulder deformation. I was at work, and I called her and I was so excited!! (yes, I was at work, but don't tell anyone that I play around with Blender when I should be doing something else.)



I had conquered the shoulders. And I had developed a mechanism based on "stretchy bones" guided by little "helper bones". This mechanism simulated the behavior of actual muscles in the shoulder and chest area. After all it was not that difficult, but I really had a bad time figuring the structure out.

Unfortunately for me, I discovered that this kind of stretchy structure could do just fine with all the other parts of the body. Actually it would do much better than the hammer-like armature I had set up for the rest of the body.

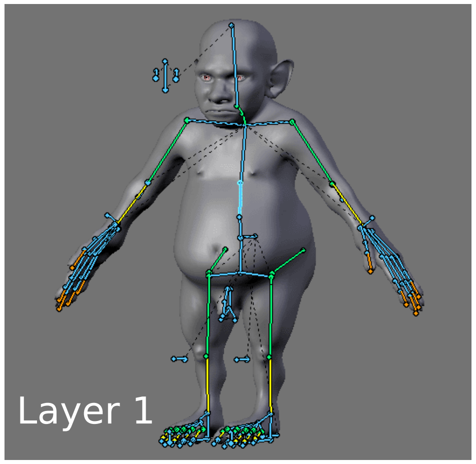
So, and this gives the reason for the name of the article, I had to rebuild all of the deforming mechanisms of the whole body according to this new technique...and that took me a while.

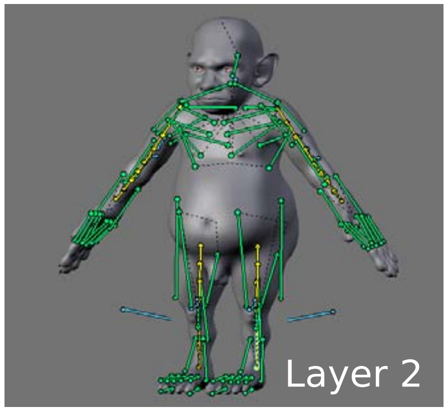
Anyway, it was worth it. And so this is **BlenRig** today.

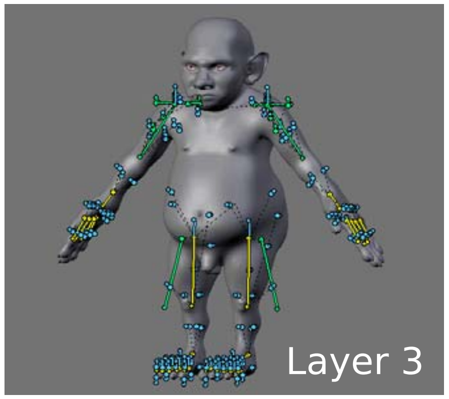
BlenRig, How it works!

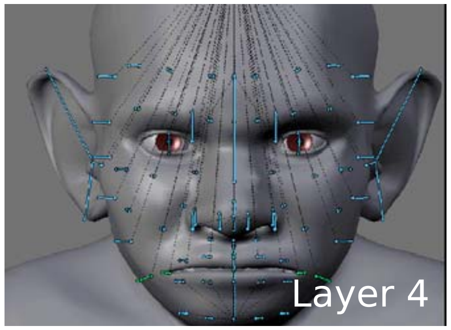
The rig is organized in 5 different layers. The main controllers are in layer 1 (body) and layer 4 (face). All of the other layers contain the auxiliary or "helper" bones and the "stretchy" or "muscle" bones.

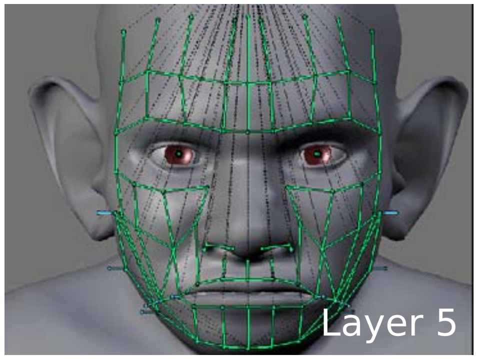
These two types of bones are not to be manipulated when animating, as they are automatically driven by the main controller bones thanks to all of the constraints mechanisms assigned to the rig.





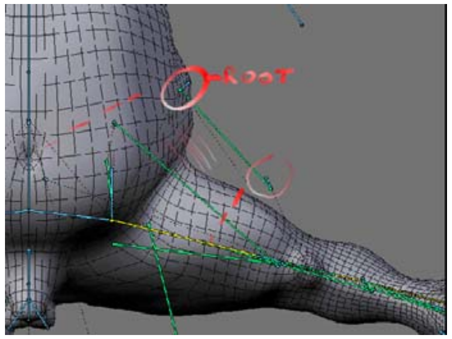






All of the rig's functionality concepts are fully described in BlenRig Manual. Here, I will try to make a brief overview on how the rig achieves the goal of producing realistic deformation on the mesh, in order to give you what I think is useful information for your rigging work in general.

The basic concept is what I was talking about when I mentioned the "Pull up" mechanism. The thing is that you have to get the vertices influenced by a force that counteracts the inwards movement produced by the joint's bending.



The best and most intuitive way I found for doing so is by creating some "stretchy" bones and placing them over the surface of the mesh. I repeat, over the surface of the mesh.

This way, you've got a helper bone linked to the bone that is root of the articulation, and another one linked to the rotating bone. These two little helper bones will drive the action of the stretchy bone. In order to make this happen, the stretchy bone must have a Copy Location constraint targeted to the first mentioned helper bone, and a Stretch To constraint targeted to the other helper.

Now, you may say that the first helper bone is not really needed, because you could just link the stretchy bone to the articulation root bone, and that is true. But having this little extra bone will allow the stretchy bone to move in space separately from the articulation bones.

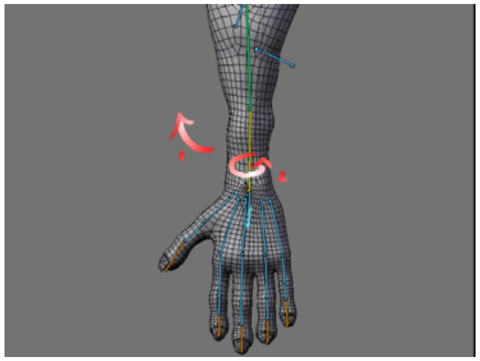
So not only does this bone stretch with the rotation of the joint, but it is also capable of transforming its location and achieving a different deformation effect (this is seen in Blenrig's chest muscles).

Let's see two examples of stretchy bone actions.

One/Two axis rotating joints

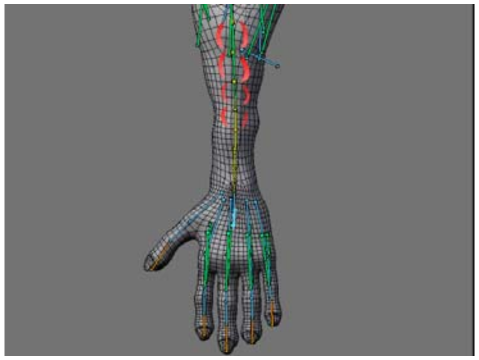
This kind of rotation is most commonly seen in the knees, elbow, finger and toes joints. They are the least difficult types of joints to fix, as they mostly require just one stretchy bone in order to achieve proper deformation.

In this example we can see the arm. The basic structure consists of two bones, Upperarm and Forearm. As you may know, this joint can have two different kinds of movement, the bending one and the twisting one. The deformation of the twisting



movement is easily solved with some extra bones placed along the upperarm. These bones follow the rotation of the forearm gradually. The closer to the forearm the more influence they get from its rotation. This way, the twisting bone that is at the shoulder point has practically no influence from the rotation of the forearm .

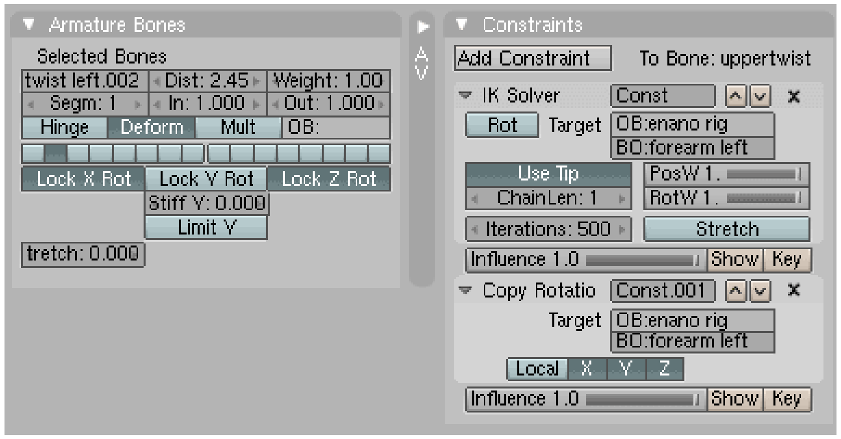
After trying many different constraints combinations, I finally found that the best solution was to parent these twisting bones to the upperarm, and then applying an IK solver and a Copy Rotation constraint targeted to the forearm. The reason why I used an IK solver instead of a Track To constraint is that with the IK solver you can easily lock and limit the axis of rotation of the bone.



This same procedure applies to the forearm in relation to the hand twisting, to the Thigh bone in relation to the Calve, and to the Calve bone in relation to the Foot twisting movement.

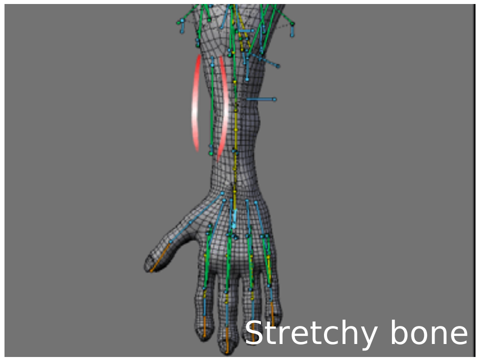
Now, concerning the bending of the arm, we have to make use of the stretchy bone.
As I said before, this type of bone has to be placed on the surface of the mesh in order to achieve the "Pull up" effect I mentioned earlier. Its correct length greatly depends on the limb's own length and on the character's joint thickness or mass.

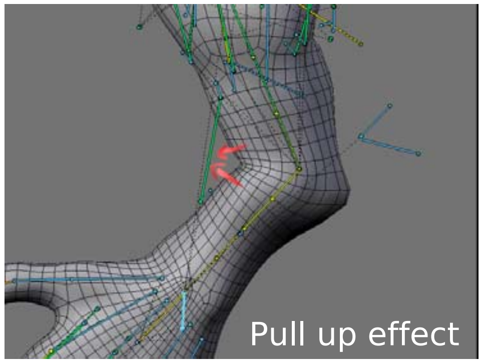
But what's most important of all is that when the joint rotates, the stretchy bone must rise over the surface of the model. Keep that in mind, because that is how the "Pull up" effect is achieved.

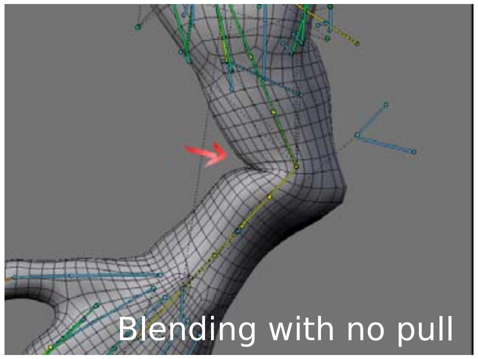


Armature Bones
Selected Bones
twist left.002 | Dist: 2.45 | Weight: 1.00
Segm: 1 | In: 1.000 | Out: 1.000
Hinge | Deform | Mult | OB:
Lock X Rot | Lock Y Rot | Lock Z Rot
Stiff Y: 0.000
Limit Y
tretch: 0.000

Constraints
Add Constraint | To Bone: uppertwist
IK Solver | Const
Rot | Target | OB:enano rig | BO:forearm left
Use Tip | PosW 1.
ChainLen: 1 | RotW 1.
Iterations: 500 | Stretch
Influence 1.0 | Show | Key
Copy Rotatio | Const.001
Target | OB:enano rig | BO:forearm left
Local | X | Y | Z
Influence 1.0 | Show | Key



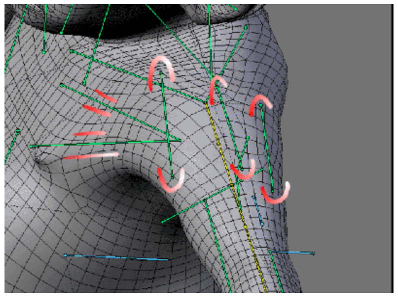


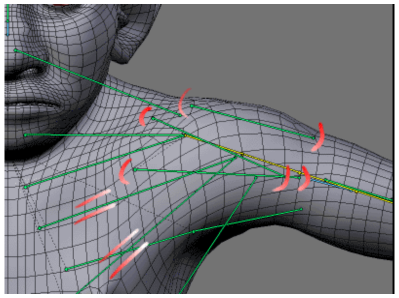


Three Axis rotating bones, Ball Joints

Well, these are by far the toughest joints to solve, mainly because deformation occurs at the four sides of the limb. This kind of joint can be found at the shoulders and at the point where the leg and the hip connect. You can also find it at the neck, at the spine, at the wrist and at the ankle, but in these cases the maximum angle of rotation is not such as for having the need of using stretchy bones.

I will take the shoulder as an example here.





As you can see, I've placed three different stretchy bones, one at each side of the shoulder. This prevents the skin from shrinking when the shoulder rotates.

In addition to this, I've placed a set of bones that roughly simulate the muscles of the chest and of the back. These bones are fixed to the torso of the character, and their tips stretch as the arm moves. In other words, these stretchy bones have a Copy Location constraint targeted to a helper bone that is parented to the spine, and a Stretch To constraint targeted to a helper bone that is parented to the Upperarm bone.

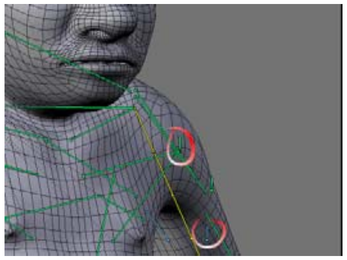
As I said earlier in the article, the good thing about having the stretchy bone copying the location of a helper bone is that the bone can move in space as it stretches. This mechanism is used in the chest and in the back bones, in order to simulate some kind of muscle bulging and skin sliding.

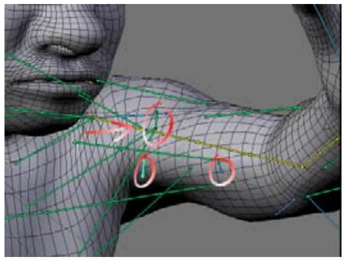
I found that having the chest bones stretching when the arm moved was not enough for getting a good and realistic deformation at the shoulders area. Therefore I implemented some secondary motion constraints to these "chest muscles". This mechanism makes the chest bones lean forwards as the arm goes forwards, and go up when the arm raises.

The basic concept of this linked structure is to have a helper bone parented to the Upperarm bone and placed near its tip, in order to give this bone a wider angle of trajectory when the arm moves. Let's call this bone "trajectory bone".

Next, you need to have another helper bone that copies the location of the trajectory bones but restricting the constraint to the Z axis. This will make this bone move forward and backward as the arm moves. Let's call this bone In Out bone.

Now, you have to parent the stretchy bone helper to this last In Out bone and you have it: the tip of the chest bone moves forward and backward as well as it follows the arm movement.





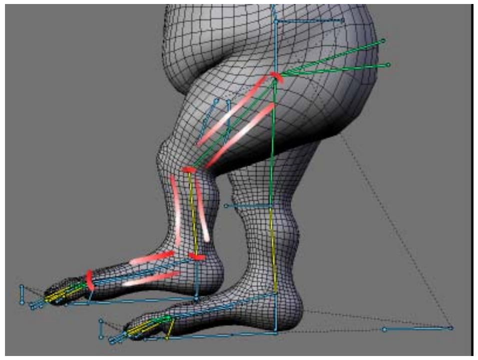
Repeat this procedure changing the restriction Axis in order to have bones going upwards and downwards, etc.

About IK's and other movement issues

Finally, I'm going to talk a little bit about BlenRig's IK system.
Getting proper IKs to the legs gave me a lot of headaches, but here's how I did it.

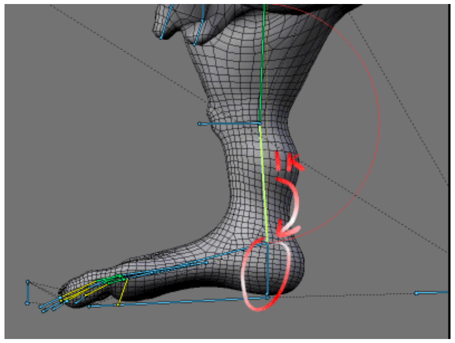
Basically, BlenRig has a double IK system that allows the rig to raise its legs when raising its feet, but also to be able to raise its heels keeping its toes on the ground.

Here's the basic parenting chain.



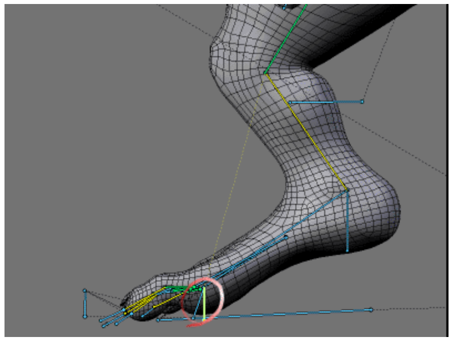
Thigh - Calve - Foot

The calve bone has an IK solver parented to the Heel bone. Thanks to this IK solver the leg can be lifted by moving this Heel bone. There's nothing strange about this one.



But now, here is the "not so easy to figure out" constraint. In order to make possible that the ankle raised while keeping the toes on the ground, it was necessary to add two extra bones. The first bone I added was a continuation of the foot bone. Then, I copied this bone and parented to the **Master bone.**

Finally, with these two bones ready, I added an IK solver to the first one, and I targeted it to its copy. This IK chain ended at the knees, not at the hip of the rig.



Afterwards I added another bone for controlling both the Heel bone and the Foot IK bone at once, and that gave me the possibility of moving the foot as a whole.

Thanks to this IK setup it is also possible to make the character crouch by moving the hip down.

Summing Up

Well, I hope that this little article was useful for you. There are a lot of other BlenRig's aspects I did not cover here, as for example breaking parenting chains in order to allow the independent movement of the shoulders from the clavicle, but it is all in BlenRig's Manual and in BlenRig's .blend file itself!!

Furthermore, there's a complete guide on how to weight paint your character using BlenRig!!

So, If you're interested, go ahead and get **BlenRig** at **www.jpbouza.com.ar**.

If you have any doubts, questions or suggestions please contact me at **BlenderArtists Forums (jpbouza)**.

Juan Pablo Bouza

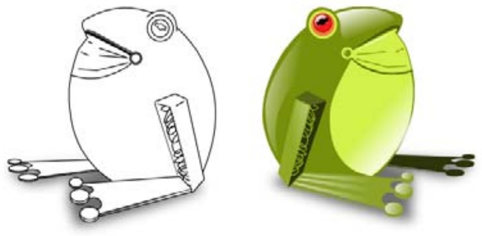
Juan Pablo Bouza is a self taught 3d Artists living in Argentina. He studied Film Making at the University of Buenos Aires, and is currently studying Music at the Art Conservatoire.

jpbouza@gmail.com

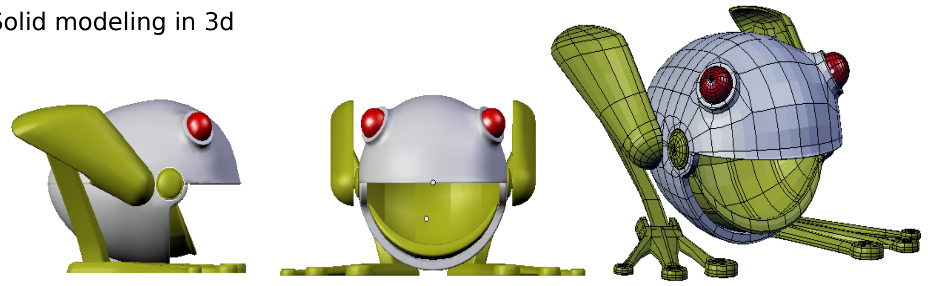
Froggy Walkthrough

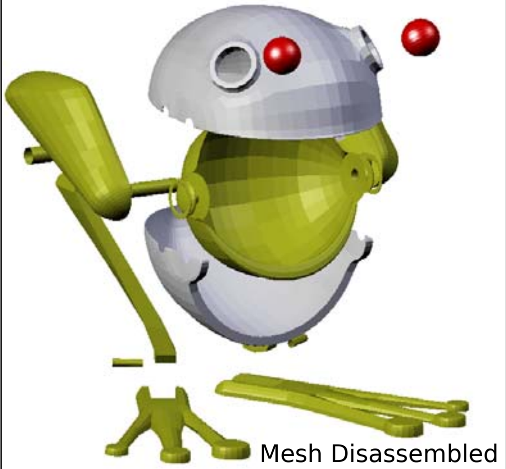
- *by* Javier Galán (chronoh)

Initial vector rendering



Solid modeling in 3d





Mesh Disassembled

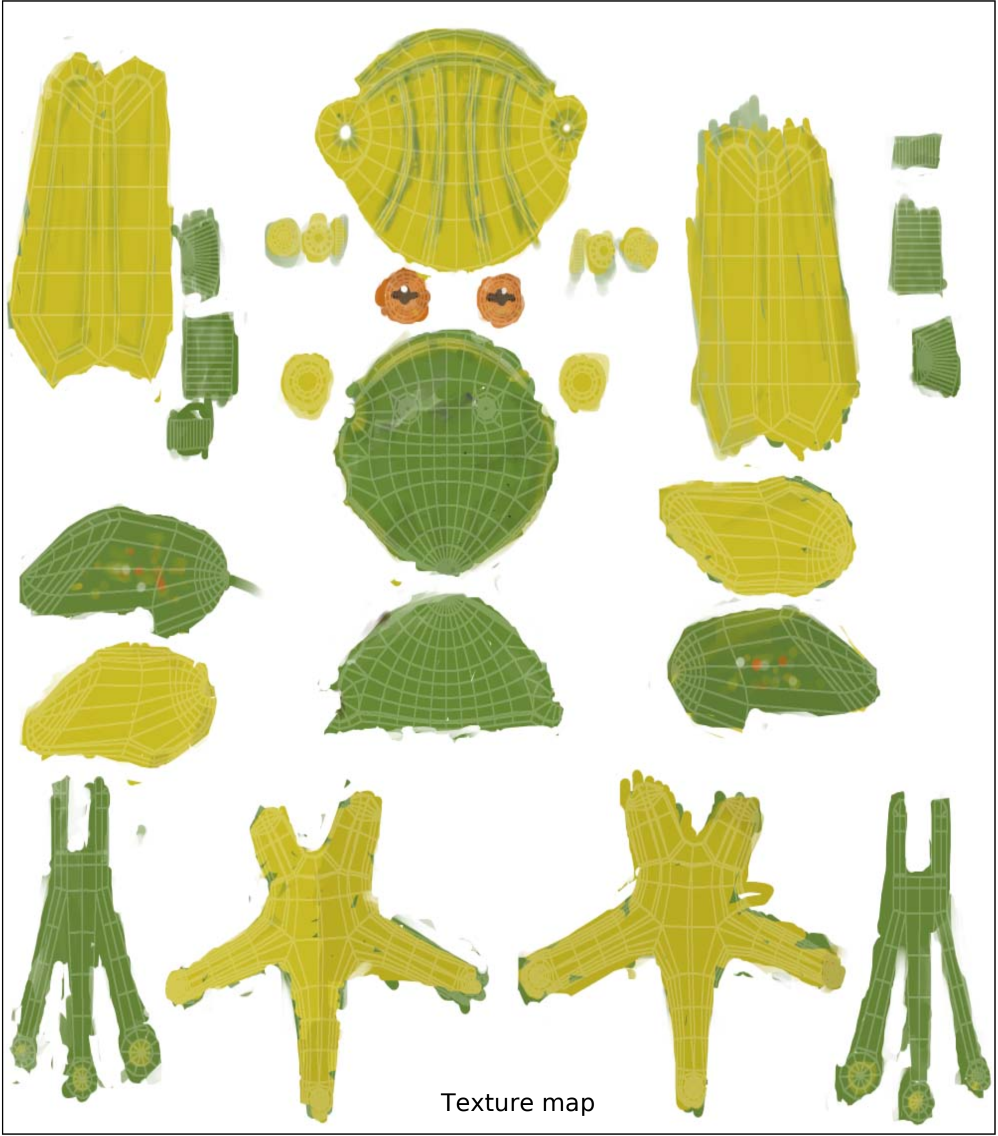
Design

After doing different 3d experiments with old tin toys, (*Zoomer* and *Awesome monkey*), I decided to put aside the idea of modeling from a real-world toy model and design my own 3d toy. Searching for some doodles in my sketchbooks, I found a little froggy that I wanted to model in 3d.

Texturing

To create the texture, I used the classic UV export script and I painted the texture in GIMP using a Wacom tablet. The texture style is very spontaneous and loose, rather than elaborate, because I decided to see what would happen after the render and post-pro phase.

For the bump mapping tests I used the same texture, desaturated (grayscale) and the RGB curves to crank up some values. This, in conjunction with the *Nor* channel, allowed to me to model the intensity of the bumps with precision.


Texture map

RenderBlog

In each scene or project, I often create a directory called "renderblog" where I put each render.

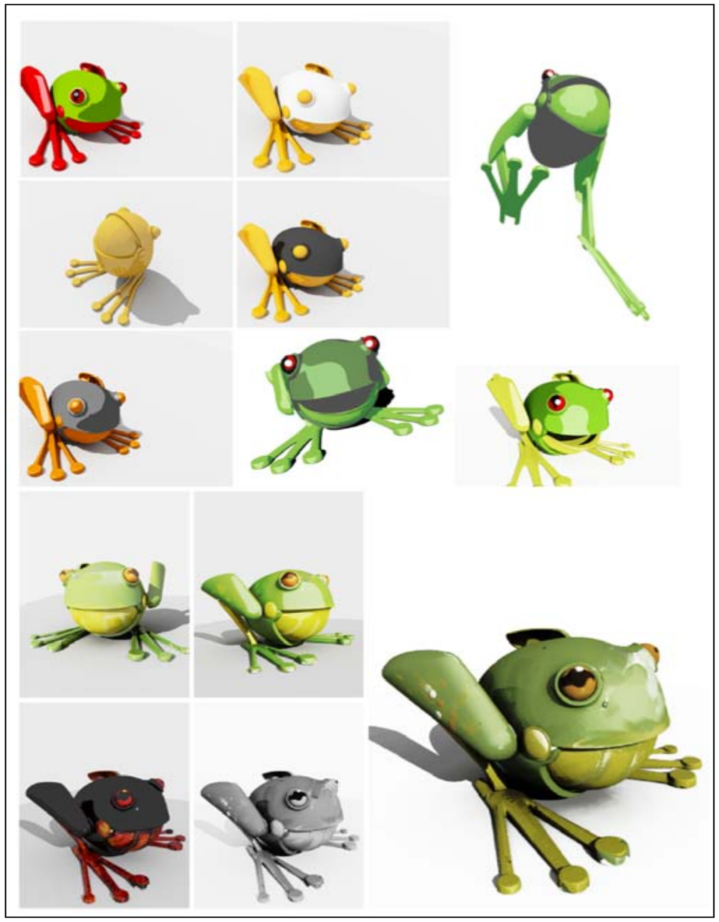
How many times have we lost an interesting situation in our scene?

Every time that we launch a render, we're investing in a time where our machine is working to generate a render. This fact becomes critical when we use external renderers such as YafRay, because GI is a time-consuming process.

Sometimes while working, we tend to dislike our render tests. But, when watching them at a later time, we start to appreciate unseen good points and other interesting results.

Another good practice consists of writing down the render settings of the interesting renders in order to reuse them later, if needed.
For the concrete case of the frog, I experimented a lot with different shader settings, both in the Blender internal renderer and YafRay, that gave me different styles.

Spending our time staring at the screen, modeling, texturing or rendering isn't a warranty that our project is necessarily evolving. In my humble opinion, it's important to often check this "renderblog" directory, because we'll have a more global vision of our progress that will let us extract the best of our tests.



Javier Galán

Javier Galán, is currently working as freelance 3d artist and illustrator from Barcelona (Spain). He tries to focus his activities in artistic and creative matters.

http://www.efedoce.com
jgalan@efedoce.com

Extinction Level Event

- *by* Jeremy Ray

Managing and Making a realistic CG movie with Blender

Extinction Level Event, (XTIN Project), is a project to do the impossible. Making a realistic CG movie with Blender is a task of great difficulty and perhaps greater unlikelihood, but that only makes it interesting. Although the project has a very long way to go to achieve its goal, it has been steadily producing art for 9 months and continues to make progress. We might just make it.

I've been asked to comment on what it takes to make a project like this happen. I'll talk about the organizational aspects, walk through the creation of our first movie poster, and then let a couple of the other guys talk about things on their end.

The Project Initiator

There's been discussion on BlenderArtists.org about how Open a Blender movie project can or should be. Some argue for a total Openness, where even the script would be a community project. I can agree with the ideology of that stance, but am unsure whether it can produce an actual finished movie. In my opinion it's more likely for a project to succeed if it's initiated by a single person with a clear vision and a high level of dedication to seeing it through.

One other quality needed is the ability to fill as many creative roles as possible, and to do so with a high level of competence. There are going to be tasks no one wants to do, tasks no one can do, and art turned in that needs a little more work to get there. The Project Initiator needs to be able to step in wherever needed to keep things flowing smoothly.

The "Pitch"

During the time XTIN Project has been active, there have been numerous attempts to pitch other movie/game projects on BlenderArtists.org. One thing almost all, (if not all), these attempts have in common is lack of preparation/poor presentation. They usually get shot down pretty quick and don't go anywhere.

A good pitch requires demonstrating that you personally are capable of doing, and have already done, work of the quality the project requires. Present a strong portfolio. If you want people to join your team, you need to demonstrate that you yourself have already put massive amounts of time into the project. There are lots of people who have a Big Idea, but don't understand how difficult the execution is going to be. And some who want to succeed by being the boss and having others do all the real work. You need to show potential team members you are not one of these, and the only way to do so is to show you've already put out.

About scripts - in artistic circles it's generally considered good advice not to get involved with a project that doesn't have a script.

XTIN Project didn't have a script when it launched, (it does now), and it survived, but there were artists who were not comfortable working that way. And possibly more who didn't join because of the lack of a script. It's better to have one than not to have one.

The "Die-Off"

Not surprisingly, most of the people who join a project under non-paying conditions will not stay.

This is not necessarily a bad thing. In the beginning the XTIN Project team had so many people that getting them organized was a full time job in itself. With a smaller team, I have more time to develop the creative side of the project. It's taking longer to achieve our goals than it would with more people, but the product should be better for the extra time put into the story/art.

Achievable Goals

XTIN Project is broken down into four, progressively more difficult, stages. The idea is that completing each stage will advance the credibility of the project and make it possible, through the increased skills of current members and bringing on new members, to tackle the next stage. The four stages are -

30 sec teaser trailer

- 1 minute short
- 10 minute short
- 2 hr movie

The plan may be changed to incorporate more steps if needed.

Financing

XTIN Project started out as a non-paying collaborative effort, but it isn't intended to remain so. I'll let Sam take over to discuss our revenue sharing plan:

Sam Rose: My name is Sam Rose. I'm an Internet social entrepreneur, interested in virtual communities, and innovative ideas related to collaborating around open technology, and "commons"-based economies. I met Jeremy Ray on the Open Business blog (http://openbusiness.cc). We were discussing different issues related to creating and distributing collaboratively-created movies and other similar content.

Jeremy mentioned that he was starting a project to create this amazing science fiction movie using Blender, the open source 3D modeling and animation application. We both had talked online about the possibilities of sharing revenue generated from a project like a collaboratively-created movie. Jeremy mentioned that he wished that someone could create an open source set of tools that resemble some of the functions of tools in emerging "crowdsourcing" projects, like Cambrian House (http://www.cambrianhouse.com/). Jeremy was looking for online applications that allow a collaborative project to create "tasks" and "roles", and allow people to share revenue from the sale of the final creative product, based on the work they contribute. I contacted Jeremy directly and let him know that I thought it was possible to make a simple infrastructure to support this revenue sharing, using existing tools and technology.

Jeremy put together a "project point system".

In this system "points" are given for each task completed in the construction of the Extinction Level Event movie. The amount of "points" a task is worth is determined by how difficult the task is. These "points" become revenue "shares". They are translated directly to a percentage of the total revenue earned by the project.

For instance, a person who does a modeling task, say, drawing a ball, earns one project point for the task. So, if the whole project only consisted of that one task, then that person would receive 100% of the revenue earned by the project. If there were only two tasks, and they were each worth one point, and completed by two separate people, those people would each earn 50% of the total revenue. However, of course there are a multitude of tasks to be done in reality. Project positions, and tasks are not limited to Blender animation, but also include eventual community marketing, accounting, and other infrastructure support. All positions and tasks will share in the revenue generated by the project.

I realized that it might be possible to create a flexible accounting application for this project using an Open Source online co-editable spreadsheet application called wikiCalc (http://www.softwaregarden.com/products/wikicalc/).

WikiCalc is very much like a normal spreadsheet, except that it is web-based. Being web based allows you to export parts of the spreadsheet as HTML or other types of code that can be embedded in different pages online. This would allow us to create a "live view" of the spreadsheet, which could be used as a report of revenue earned for project participants. wikiCalc can also export a tab-delimited text output, which can incidentally be imported into PayPal to "mass pay" revenue to project participants (using PayPal's Mass Pay function). In fact, this is how our current revenue sharing system for Extinction Level Event actually works. We will be selling content through a Drupal (see; http://drupal.org) web store, and then exporting data about revenue from the Drupal website and into wikiCalc.

For the long term, in collaboration with open banking and finance think-tank BarCampBank (http://barcampbank.com), we are developing a "peer to peer" revenue sharing and money pooling application that is less dependent on gateways like PayPal, that can use as many different payment gateways as possible. This is unfolding at http://www.wikiservice.at/fractal/wikidev.cgi?EN/BarCampBank/P2PMoney in collaboration with people all over the world interested in developing an open set of web tools that can let people easily and directly pool money, and share revenue. Lessons learned from the research and development of this application will be applied directly to the revenue sharing in the Extinction Level Event project.

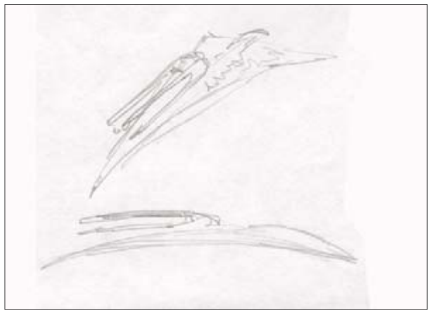
Section II

First Product - Pyramid Builder Gunboat Poster

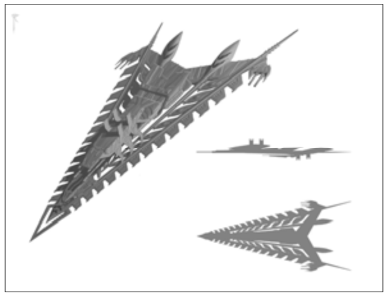
I - Concepting the Gunboat and Dragons

One of the things I've been keenly interested in developing is the visual look of the XTIN universe. In recent years I've become increasingly dissatisfied with the design work being done in movies. Although XTIN Project doesn't have the manpower to build truly intricate models, I feel like we can gain an advantage by having a unique style and designs that are superior in their broad shapes (if not their details).

I usually will go through dozens of chicken-scratch quality sketches trying to get the overall shape of a design right. Details are easy, coming up with an appealing overall shape can be difficult. Luckily this was one that just popped into my head.

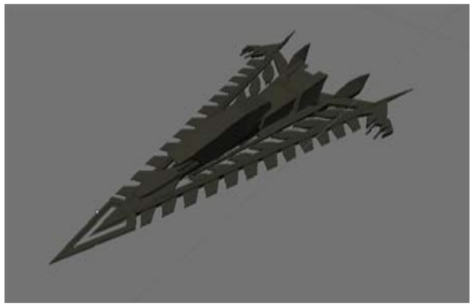


From there I made top and side views for the modeler. I used a perspective transform on the top view to create a base for the detailed painting. The perspective ends up a little off this way, but it's fast and fools the eye well enough.

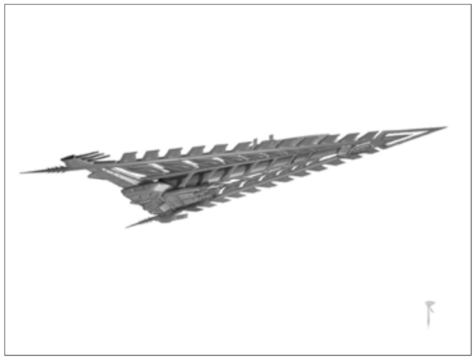


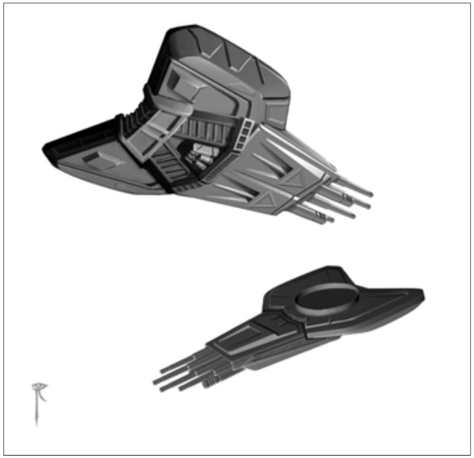
II - Modeling

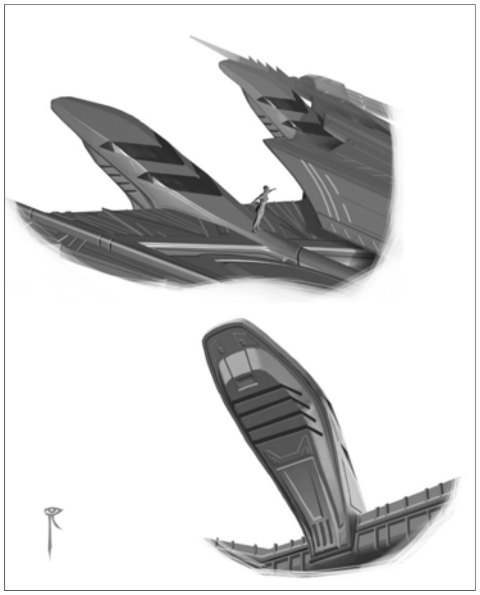
After this I passed the art to the modeler, Zach Goldstein. I asked him to make a rough version of the model first, so I could check the proportions before he detailed it. It is something of an eye opener to see how many ways a piece of concept art can be interpreted.



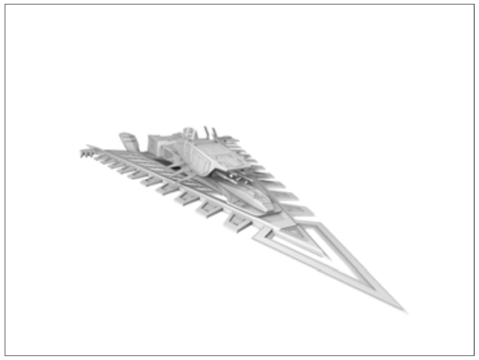
After a few changes I sent the lo-res model back to Zach for final modeling. Along the way Zach asked for additional concept art, which I provided by taking screen caps of his model and painting in the new details.



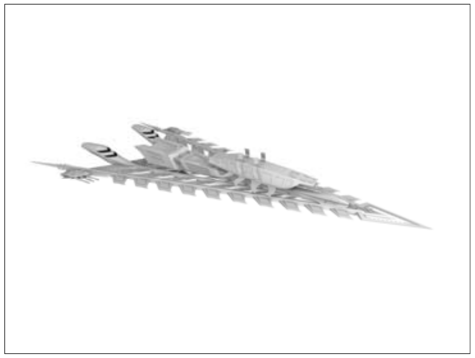




Zach finished the models.



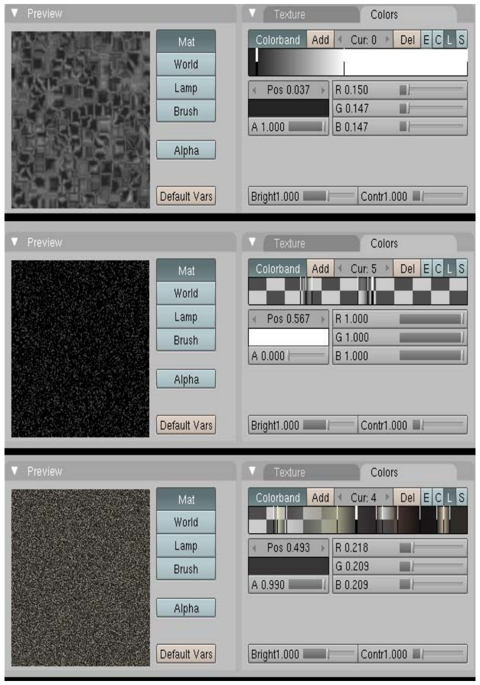
And I added few more details.



III - Texturing

Texturing and materials have been a big problem for XTIN Project. While XTIN Project has had no problem finding modelers, nobody wants to texture. Since I'm the only texture artist, I've fallen back on using procedurals as much as possible.

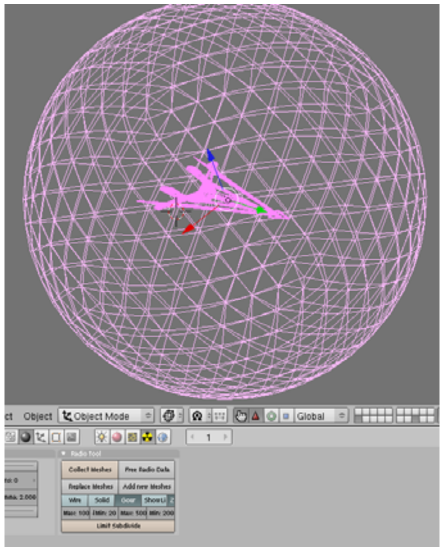
You can download a sample model here. Gunboat shader .blend file



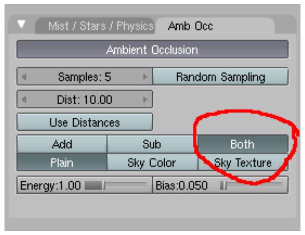
The big problem with procedurals is the lack of area-specific detailing, such as dirt gathered into corners or worn edges on paneling.

I found the solution to this problem is to use a mask for additional layers of dirt procedurals. UV mapping is usually time consuming, but in this case it wasn't necessary to eliminate all the texture stretching. All the visible dirt detail comes from the procedurals, which don't stretch even though the mask controlling where they appear may be stretched and distorted. I used unwrap (smart projections) to quickly generate UV coordinates for my gunboat sections.

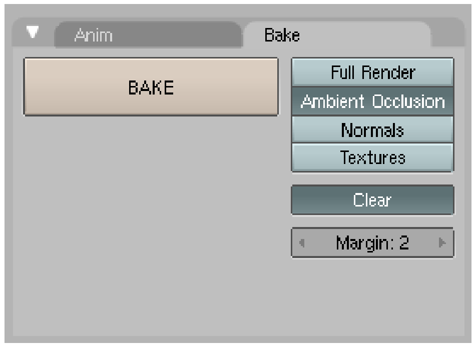
To automatically generate my dirt masks, I put each section into a spherical emitter and gave it a quick ambient occlusion pass. It isn't necessary to let the full calculation run, a few seconds will do the job.



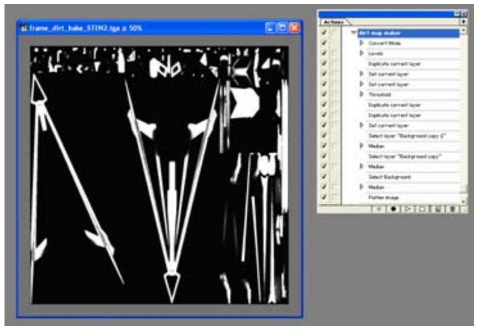
Under the ambient occlusion tab, select "both." This should give more contrast to the mask.



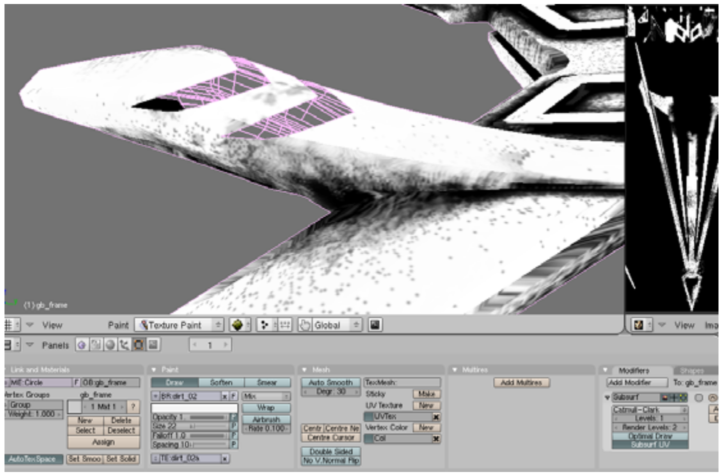
With a new image applied to the object, (in the UV/Image Editor), I used the "bake" options in the render section to extract the AO information to texture.



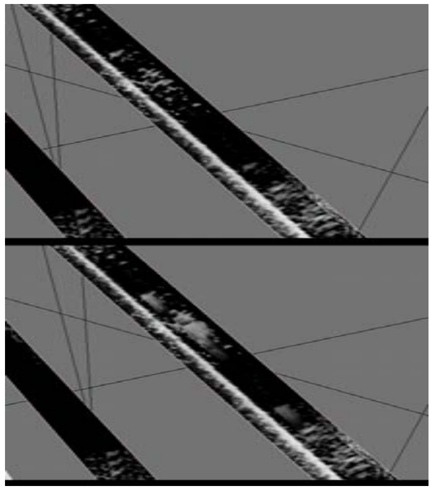
This is just the beginning - there's not enough contrast. The next step is to take the AO texture into Photoshop/ GIMP, increase the contrast and enhance the shadow/dirt areas. This is a slightly involved process which could use its own tutorial.



The last step was to use Texture Paint for a final editing of the AO texture, and to add additional dirt areas.



The smear tool is useful for wind-swept dirt streaks.

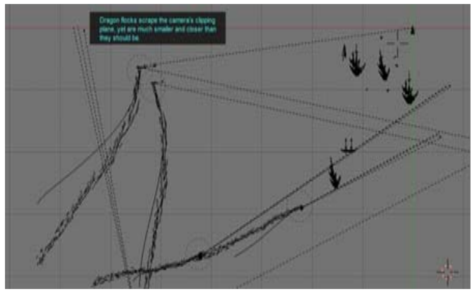


IV Scene Set Up

Scale is a problem for XTIN Project for which there does not appear to be a good solution. Between the values of "clip start" on the spot lamp and "clip end" on the camera, it's not possible to have objects ranging from people size to pyramid size all in one scale. Aerial battles between capitol ships vanish beyond the range of clipping planes.

Faces can't be lit by consoles because the clipping start for the lamp is on the other side of the head.

For this scene I was able to solve the problem by reducing the scale of things that would be in the distance and putting them closer than they would be.


Dragon flocks scrape the camera's clipping plane, yet are much smaller and closer than they should be

As this is a still image the solution works. If the shot were animated, with dragons swooping in from the distance and their color affected by atmospheric haze, it wouldn't.

V Compositing

The scene has over 10 million faces. It uses three Blender scenes, seven render layers and one image layer for the matte painting.

Blender's compositor was used primarily for distance blur and layering the render layers together to create the final image. It was very useful for adding glow effects, although a dedicated glow node would be a nice addition.

Most of the actual compositing work however was done in Photoshop. At 5400 x 2433 resolution, Blender's compositor crashes often. At a lower resolution it might have been possible to add more nodes to the scene and use Blender's color nodes.

Animated Castle Effect Walkthrough

- *by* David Revoy

Introduction

After finishing my ‘Little Fairy’ animation, I received a lot of questions about how I created this animation. This little walkthrough answers to the most commonly asked questions. Many thanks for all the comments and e-mails!

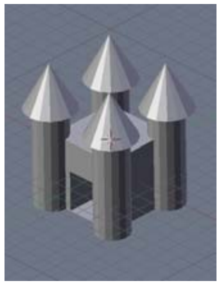
1) How did you create the castle effect?

Here is a tutorial to explain it from A to Z. Upon completing this, everybody will have knowledge of halo material, and will be able to animate a nice magical appearance for an object.

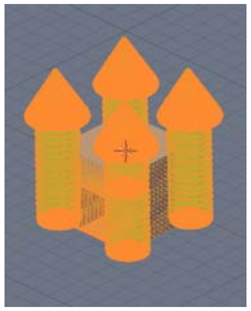
Model And Material

1) Building a simple castle with some primitives:

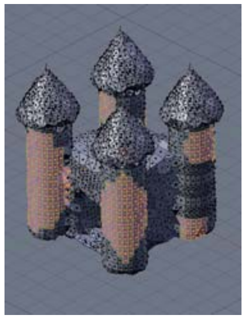
- A cube with a door simply extruded
- 4 cylinders for the towers
- Customize a cylinder to make a roof by selecting all the top points and welding the vertices together with a high value of Remove Doubles, don't forget to re-center the center point.



2) Join all of the primitive objects two by two with Ctrl+J, (i.e. - select two objects, the tower and a roof, press Ctrl+J, and they're joined)

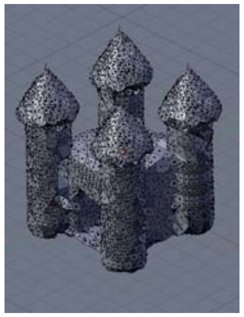


3) With your castle mesh selected, toggle into Edit mode (Tab-key), and subdivide 4 times. (W-key, subdivide, repeat the process 4 times)



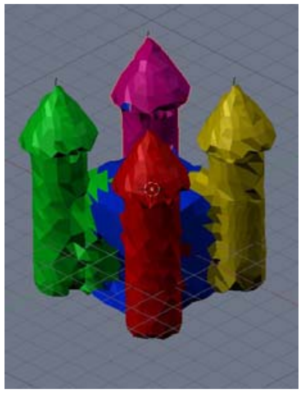
4) Keeping your model selected, go to the Edit panel and change the "Remove Doubles" value to 300. Press the Remove Doubles button; it will remove vertices leaving a beautiful noise pattern. Press the Smooth button 4 times, and Remove Doubles again.

5) Now with Paint Selection (press the B-key 2 times), select the zone with too many square polygons. Remove Doubles with a value of 400, re-subdivide. You can continue this process until you arrive at a good fractal point on your object.

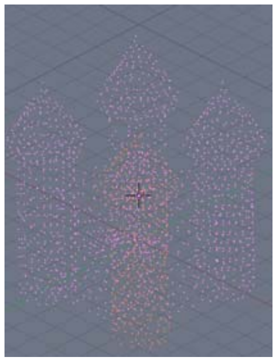


6) In order to prepare the animation effect, divide your shape into objects. In wire frame mode, select each tower from the Top view, and press the P-key to separate the selected object from the mesh.

Now, you should have 5 objects. (See the picture below)

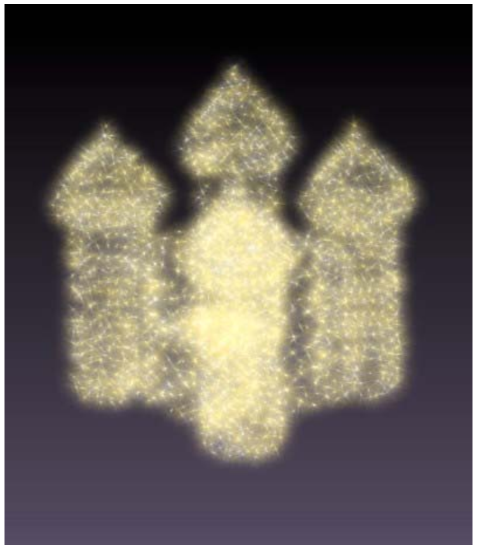


7) Add a material to one part, configure it as shown in this screen: Assign this material to the other part.



8) Before rendering, go to the World tab to change the default blue background screen. Replace it with a nice Paper/Blend combination using desaturated violet to Black.

9) Done! A beautiful castle.



FX for Animation

For each part of the castle, add a "Build" Modifier with the Randomize option selected and a length value of 200. You can now control the start and the end of each part of your building. (If you break your model into more parts, with more parameters, you can have the result of your choice.)

For faking the explosion of particles that builds the castle, at the center of the castle, create an Icosphere with a subdivision setting of 2. Give it the same Halo material as the castle. Go to the Physics Buttons and add a Particle Effect. In the Particle Motion panel, increase the Normal attribute to 200, add a Random setting of 1, and change the Particle Number to 500. Make the Start of this animation at -10, and the End at 80.

Render the frame with your chosen video format settings.

Note: In order to fully appreciate the volume and depth of the appearing castle, this effect is best viewed from a camera in motion.

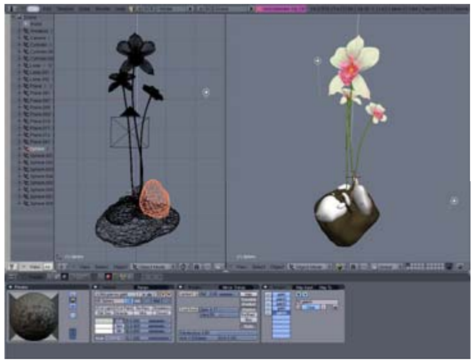
1) How did you make your water effect?

During rendering, water was the most time-consuming part of my animation. Typically, my water is just a 100% pure raytraced mirror with a cloud texture applied on the Nor channel of the material. It was animated with an offset in the Map Input panel.

2) How long did this animation take to render?

At the start, I tried to render my elements as fast as I could,(I only work with the internal render engine, because I have gotten to know the quirks of the internal render engine and how to achieve fast renders). I limited myself to a 2 min. max/frame.

I rendered my scenes during the night. None of my scenes exceeded 300/500 frames, so it was possible to work on the animation during the evening, render through the night, and repeat as needed.



3) Why did you choose a Fairy theme?

I chose to do an animation on fairies, because I really like little fairy's. They are everywhere in my galleries. When I started this project, I wanted to create a fairy animation to show two little girls in my family. Originally, I had a story with a moral, but I finally decided to just show it as actions, a suite of beautiful scenes without trying to really tell a moral. Stories that say what should be or not on TV always bore me. The result was a documentary type look at a fairy, but with an intimate view, as if we were sharing how her morning starts.

4) How did you make the ray of mist from the top of the grotto?

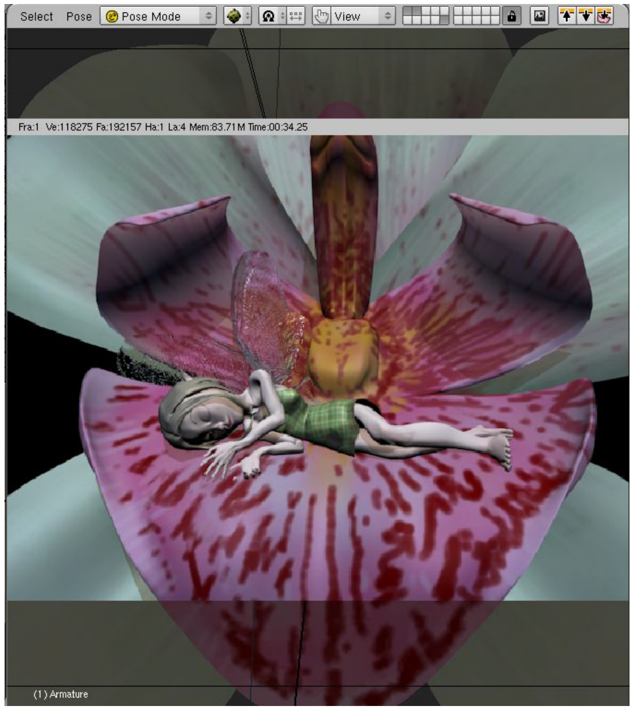
It's a fake mist; just a Cylinder with a big white Halo material, with an Alpha. I have produced clouds with this technique; it is simple, easy and fast to render.

5) What was the most useful thing you learned while making this animation?

Render animations using JPG files set to 100% quality. Sometimes the computer crashed, because of electricity going out or just because of the rendering itself. By rendering individual frames, you don't lose a whole night's rendering. Also, rendering individual frames allows you to make corrections to frames that contain artifacts. Additionally, having all the frames in the JPG format is economical for disk space■

Voilà :-)

David Revoy
http://www.davidrevoy.com
info@davidrevoy.com



Fra:1 Ve:118275 Fa:192157 Ha:1 La:4 Mem:83.71M Time:00:34.25

Hello again and welcome to our second installment of "Meet the Blenderhead".

In this issue, we meet a user who utilizes Blender in his daily workflow for low-poly character modeling for the game industry. You know him best on the forums as Eldron. Jonatan Poljo's mastery of low-poly modeling and his awesome skills in sculpting and texture painting have made his work instantly recognizable throughout the Blender world.

Enjoy!

Derek Marsh (BgDM)

1) You have been using Blender for some time now. Can you describe how it fits into your daily workflow?

Well, using Blender at first was high-poly modeling for me, which later progressed into low-poly modeling and then, the game art I've done now.

Fortunately enough, knowing high-poly is exactly the thing needed to do what you do with game art today, normal map generation.

I build the base mesh that I sculpt from, can be a simple cube to a more complex mesh depending on what the high-res is supposed to look like.

The sculpted high-poly is then imported into Blender and decimated to a workable poly count. I create the low-poly model around the high-poly and unwrap it. The high-poly and low-poly are then taken into 3DStudio Max for normal and lightmap baking.

I have been trying to get back into regular high-poly models for rendering inside of Blender though, since I find it quite interesting and I love seeing what people create with Blender, but never get the time to actually make a whole rendered scene.

2) With your experience using Zbrush and Mudbox, do you see the current Sculpt Mode being usable in the near future for high levels of detail and normal/displacement map generation?

The sculpt mode in Blender follows closely to the ease of that in Mudbox, which makes it very easy for anyone to pick up.

However, I think that sculpting is really an area of its own, just like how making textures is, and even though Blender is quite a good tool for a lot of things, I don't think it can beat a specialized tool like Mudbox, Zbrush, or Photoshop in the case of texturing.

It mainly comes down to how fast Blender can be at pushing polygons when you want as many as possible when sculpting something big and detailed, and it doesn't come close to handling the insane amounts you reach when sculpting something in Mudbox. However, speed aside, it is a great tool, and I see no problems with creating high poly assets for normal map generation with it, hopefully it'll be a completely seamless process when Blender gets proper normal map baking, and with the tool itself becoming faster.

3) Being an artist employed in the gaming industry, have you had any success in converting anyone using other commercially available applications over to Blender? If so, what has their reaction been?

People like to stick to what they enjoy the most, and in the end have to learn what they are required to use at work. So, if some guy sits with Maya and enjoys it, he's not going to be easily switched over to Max or Blender, nor the other ways around.

They are all equally strong tools and it all comes down to what people prefer.

I have, however, always experienced that people have heard about "this Blender application", so it has definitely left its mark.

4) Is there a feature that you would like to see added to Blender?

As far as I know, lots of the features I'm looking for are being worked on right now. But one thing would be a faster viewport, since Blender development has always felt to me like having faster rendering in mind, but as a game artist, I'd just as much want to have faster handling and viewport realtime rendering of models.

5) From your working experience, is there anything that you can share with people looking at getting into a gaming house or professional CG house?

Practice, a lot, but that's a given.

More importantly, never be completely satisfied with what you do because then you won't see where you have to improve, always look for ways to get critique, and to have the things you work on be seen by as many eyes as you can get them to. If it's game art, spend a lot of your time at the Polycount forums, (boards.polycount.net), many there are brutal and honest industry professionals and can give you the most valuable feedback you'll ever find anywhere. And besides, if you do behave nicely and not act like a jackass on the multitude of CG-related forums around the web, you might actually get to know people in the industry, and that is usually the one break a junior artist can need to get his foot inside the industry.

Take in what you learn, keep doing new stuff, redo the old, it takes time to get a hang of it, but when you do learn new things you'll enjoy the feeling of it.

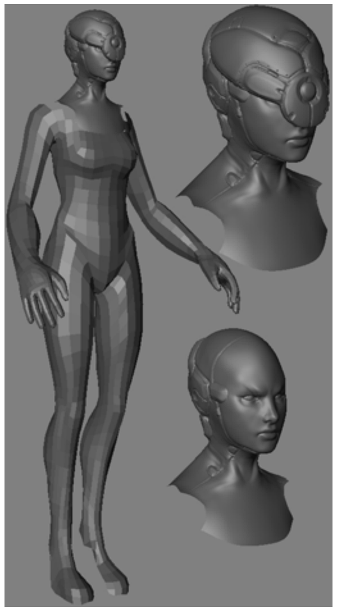
Drawing or taking up sculpturing when you don't feel like modeling is also another thing that helps you improve.

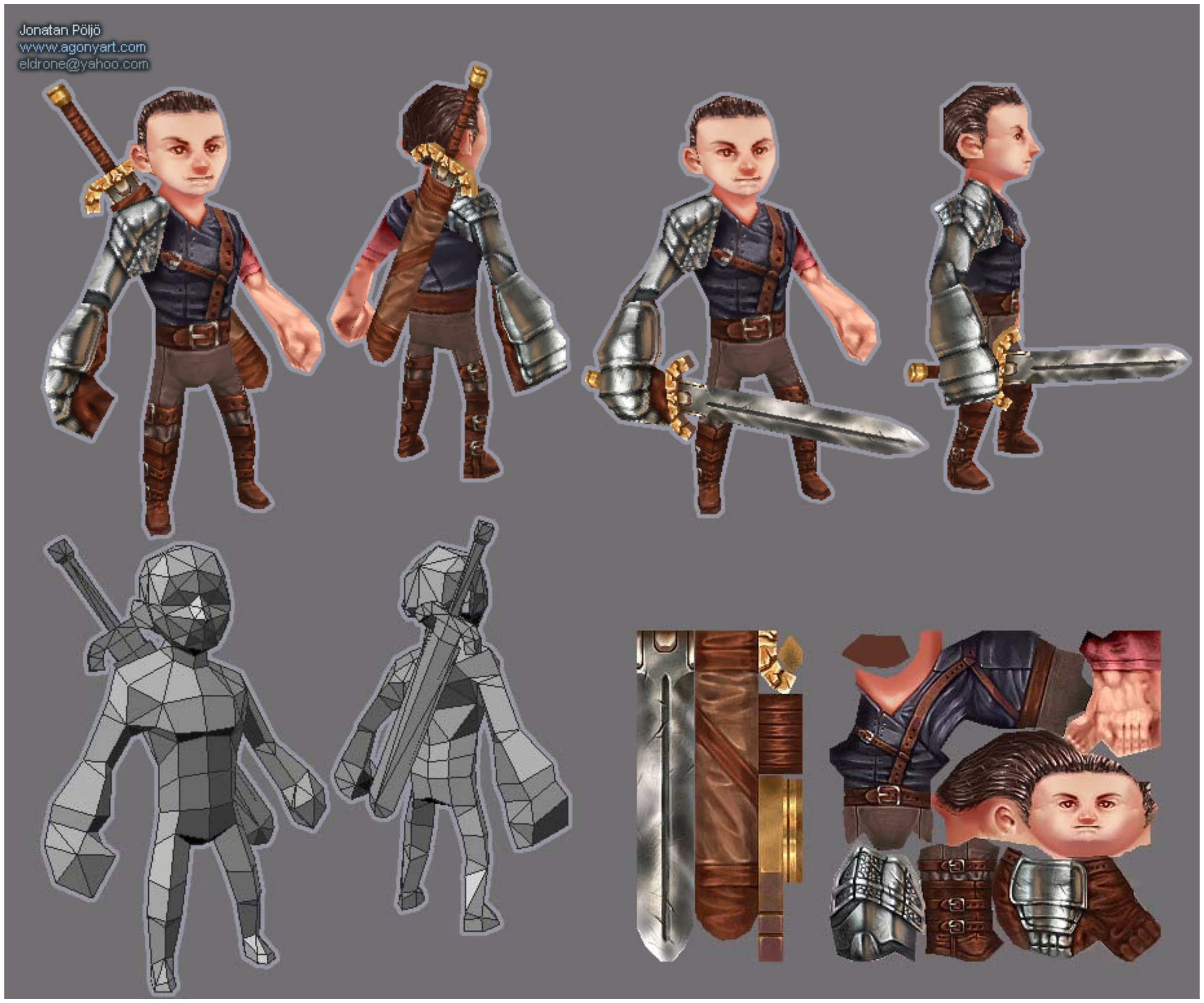
And when you do get in, remember to keep updated, learn new tools, learn new methods, be flexible.

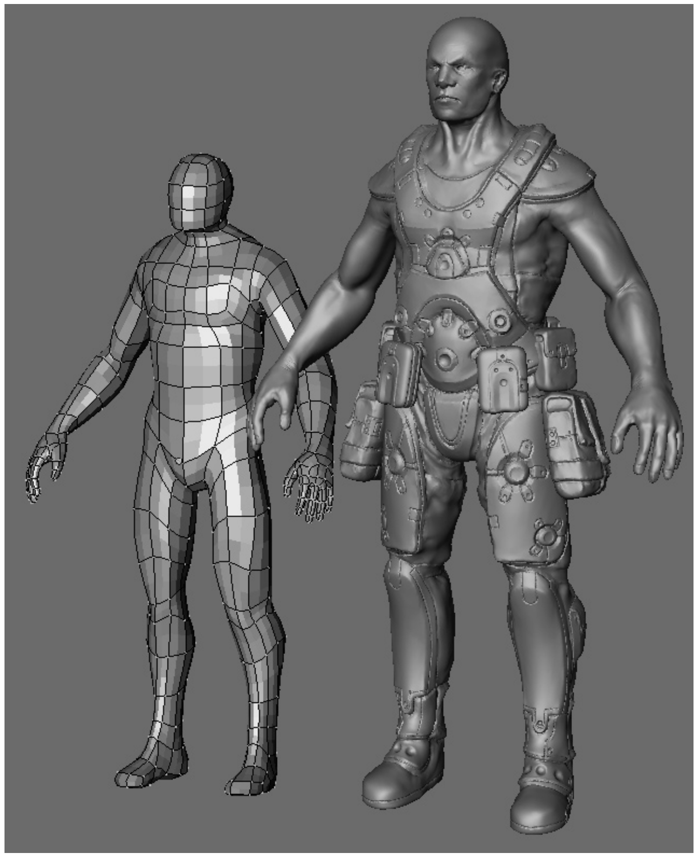
6) Where do you see Blender progressing over the next few years?

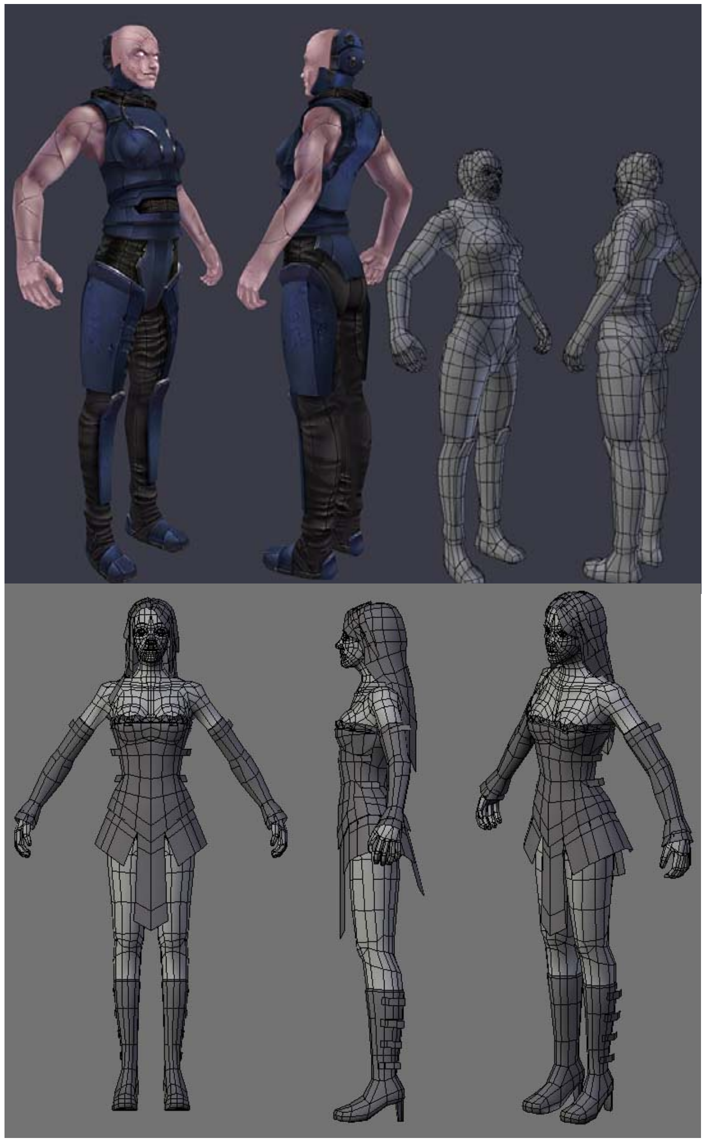
I see Blender slowly but surely growing in popularity as it has, and more users becoming experienced enough to get jobs in various industries while using Blender, thus feeding the promotion of Blender even more.

With Blender itself, we'll probably start seeing it gain features faster than most professional applications can add them ▪




Jonatan Pöljö
www.agonyart.com
eldrone@yahoo.com





Interested in writing articles for BlenderArt Magazine?

Here is how!

1. *We accept the following:*
   - Tutorials explaining new Blender features, 3dconcepts, techniques or articles based on current theme of the magazine.
   - Reports on useful Blender events throughout the world.
   - Cartoons related to blender world.

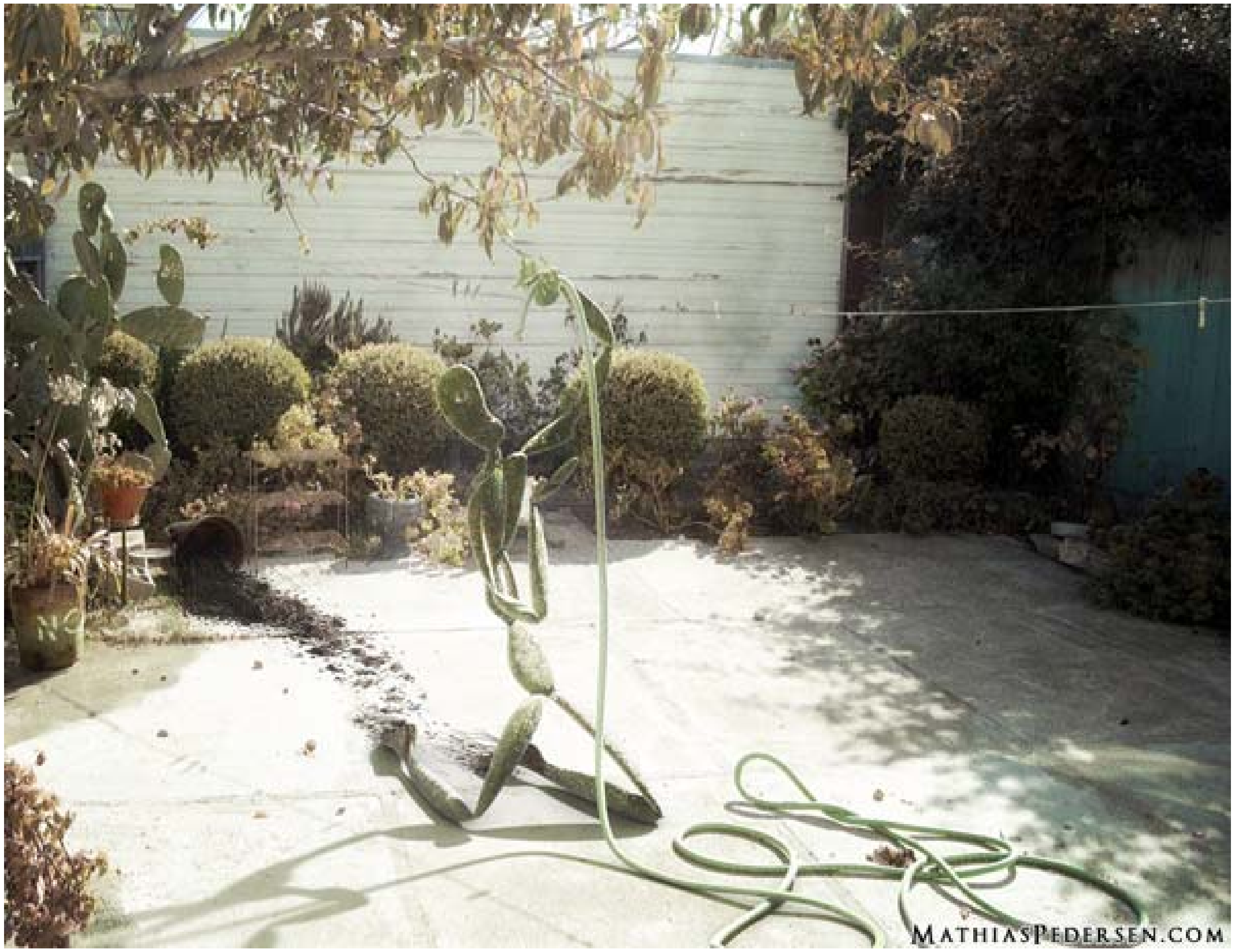
2. *Send submissions to sandra@blenderart.org. Send us a notification on what you want to write and we can follow up from there. (Some guidelines you must follow)*
   - Images are preferred in PNG but good quality JPG can also do. Images should be separate from the text document.
   - Make sure that screenshots are clear and readable and the renders should be at least 800px, but not more than 1600px at maximum.
   - Sequential naming of images like, image 001.png... etc.
   - Text should be in either ODT, DOC, TXT or HTML.
   - Archive them using 7zip or RAR or less preferably zip.

3. *Please include the following in your email:*
   - **Name:** This can be your full name or blenderartist avatar.
   - **Photograph:** As PNG and maximum width of 256Px. (Only if submitting the article for the first time)
   - **About yourself:** Max 25 words .
   - **Website:** (optional)

Note: All the approved submissions can be placed in the final issue or subsequent issue if deemed fit. All submissions will be cropped if necessary. For more details see the blenderart website.

the silent killer
MATHIASPEDERSEN.COM

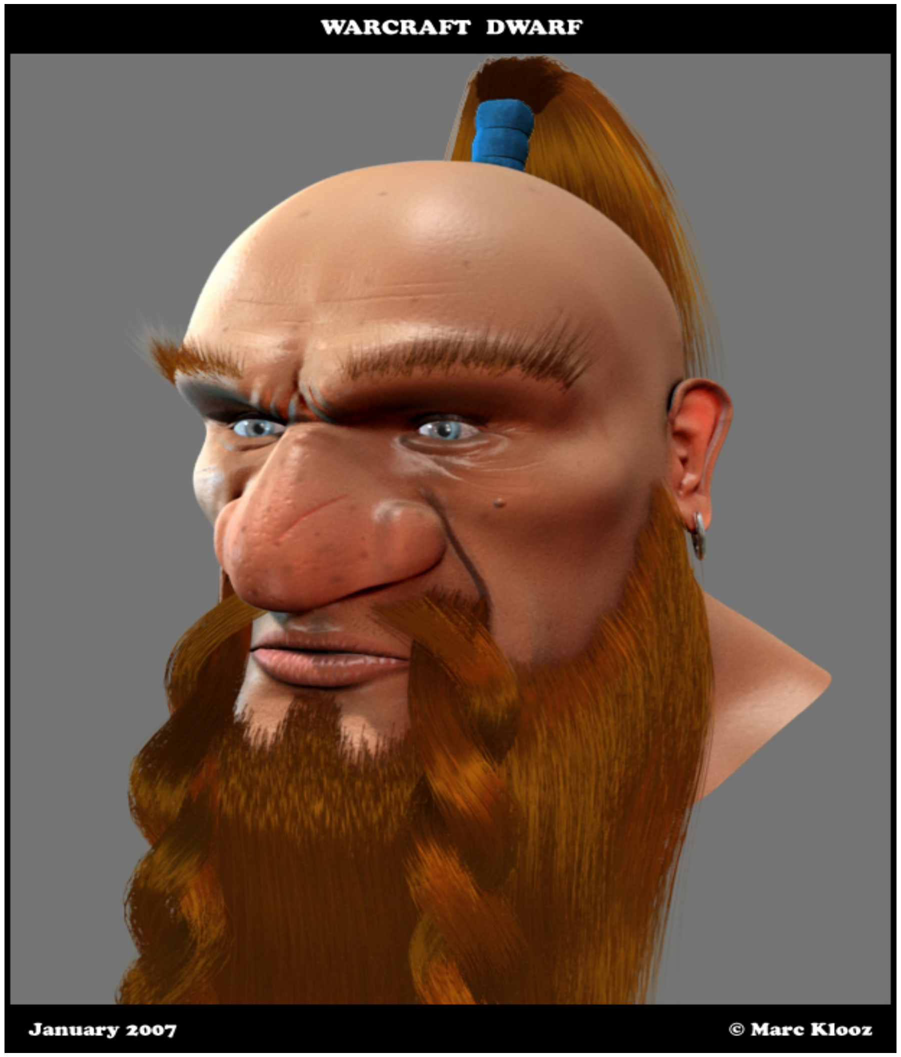
Silent Killer - Mathias Pedersen



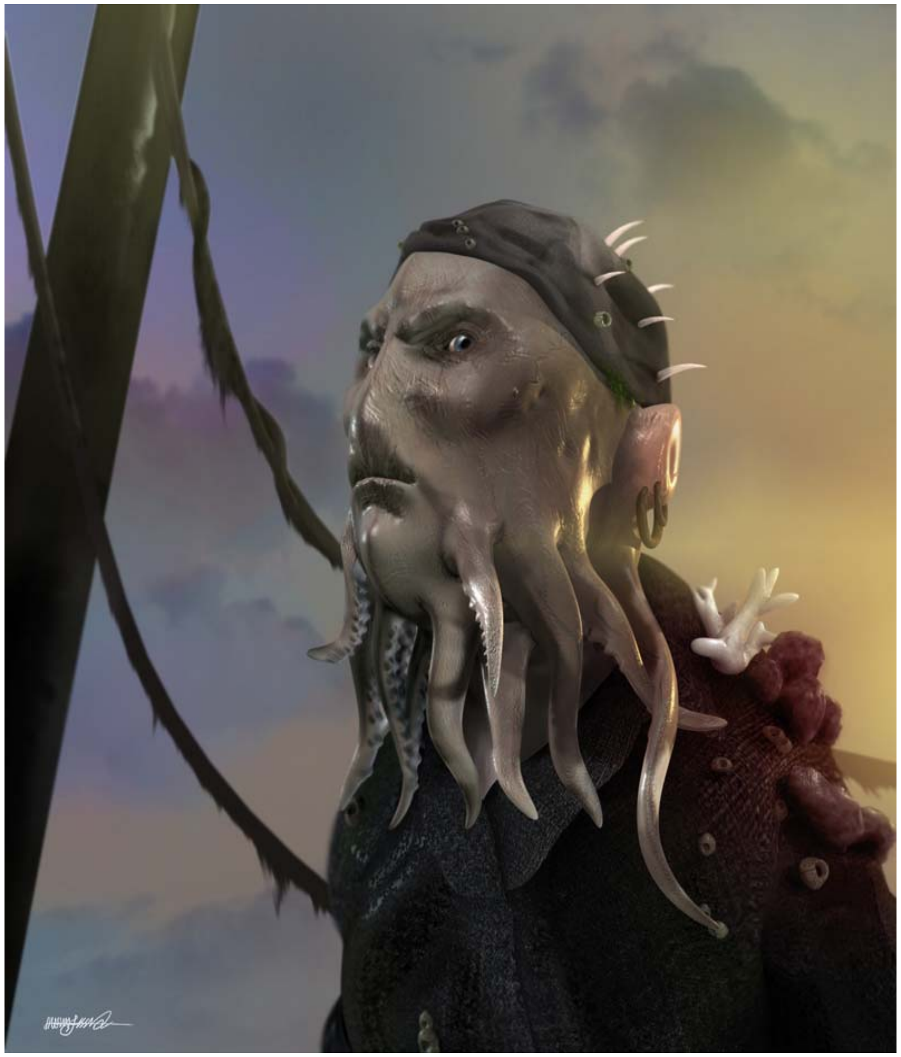
Thirsty - Mathias Pedersen

opedroaugusto@hotmail.com - based in the drawings of kidcomics.com
opedroaugusto@hotmail.com - based in the drawings of kidcomics.com

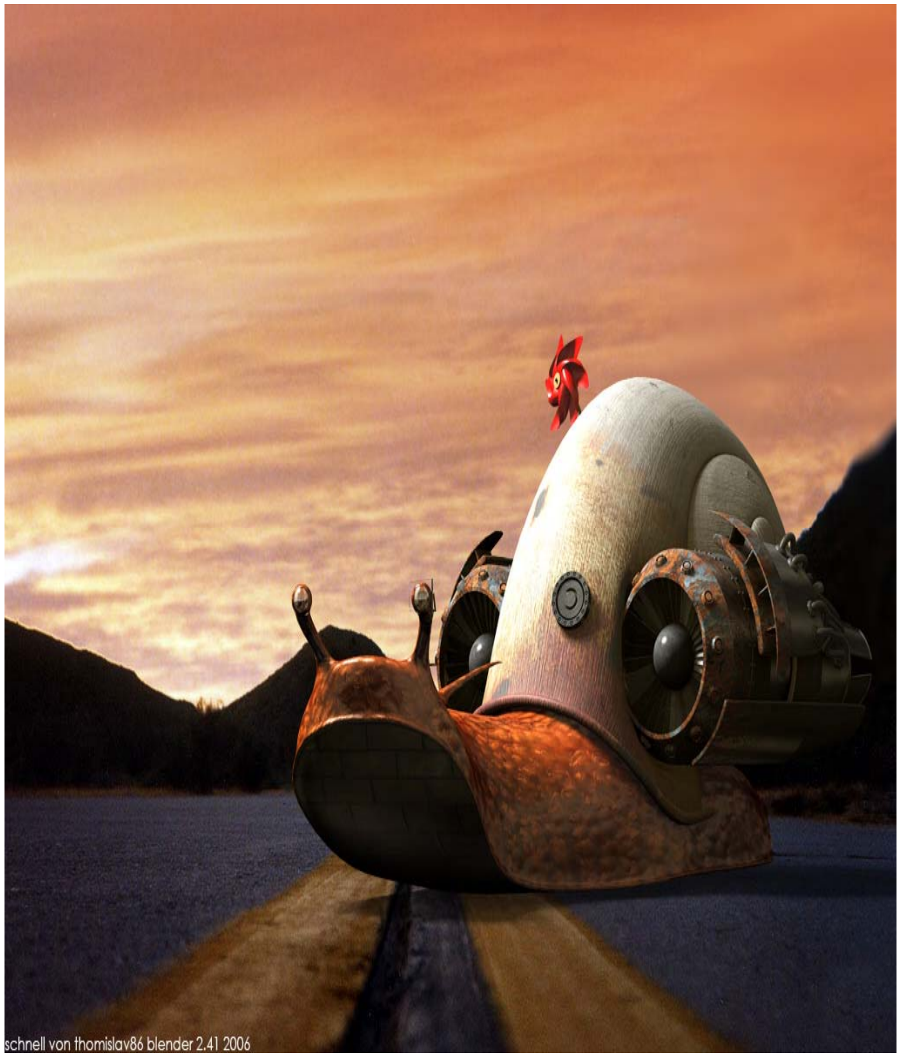
Cartoon - Pedro Augusto



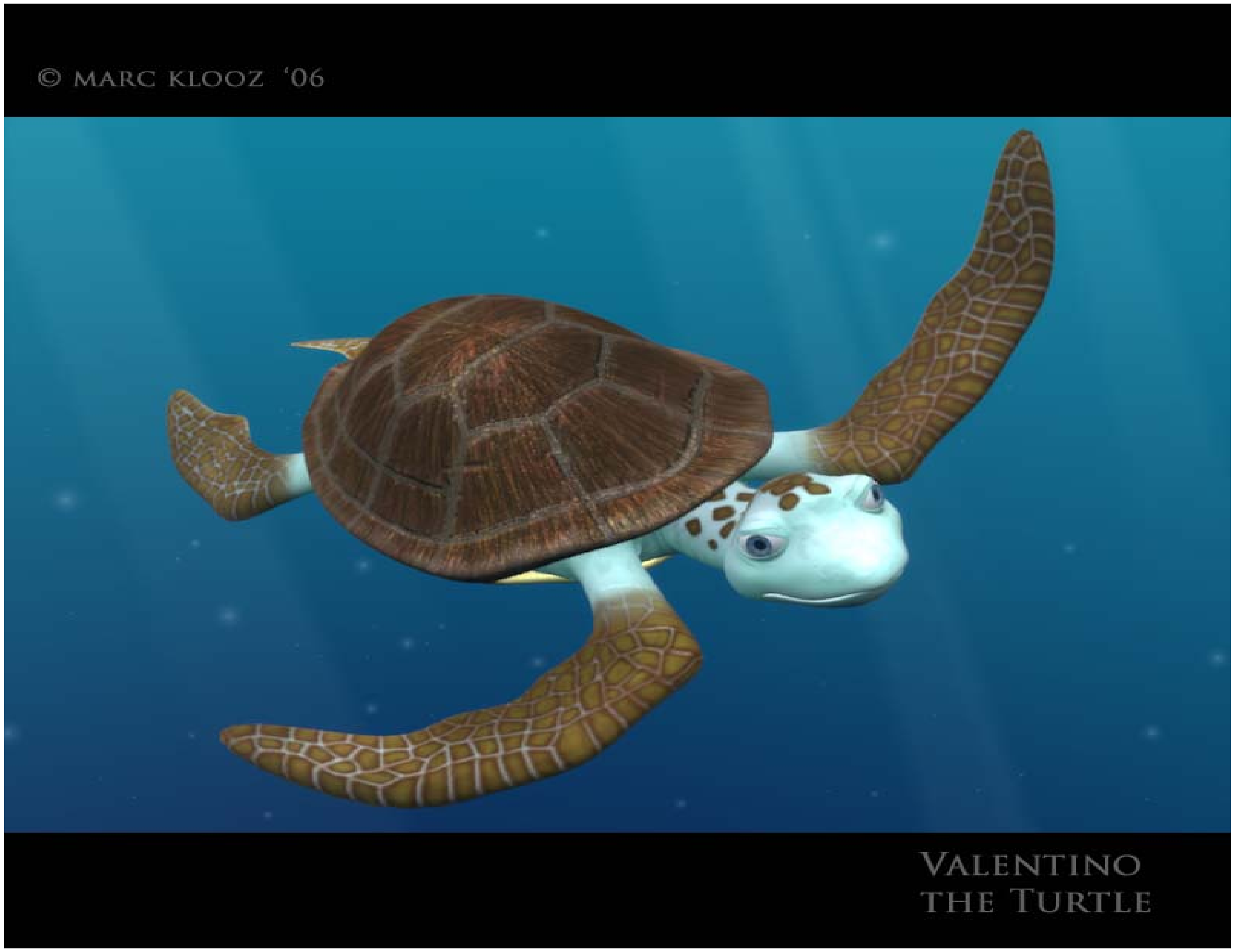
Dwarf - Marc Klooz



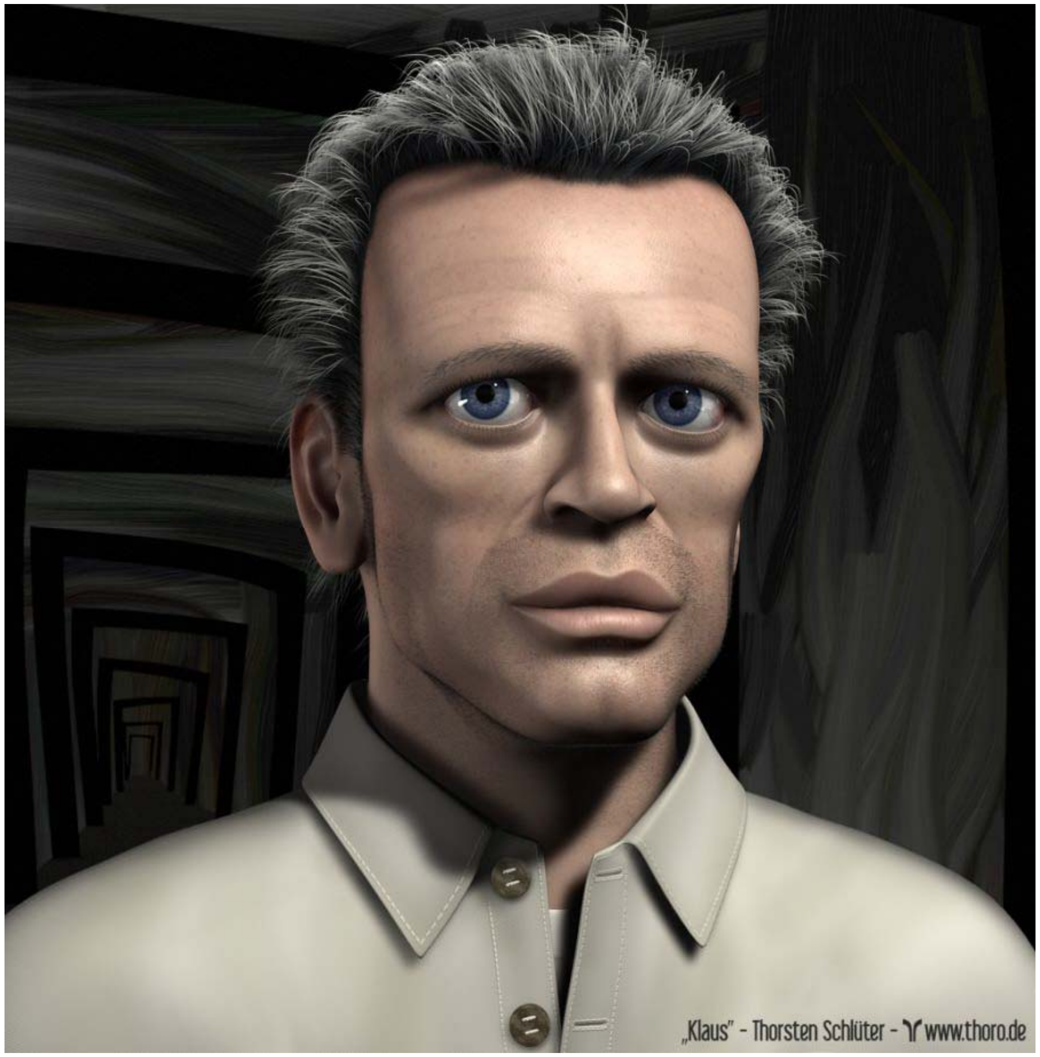
PunkyJones - Al-Haitham Jassar



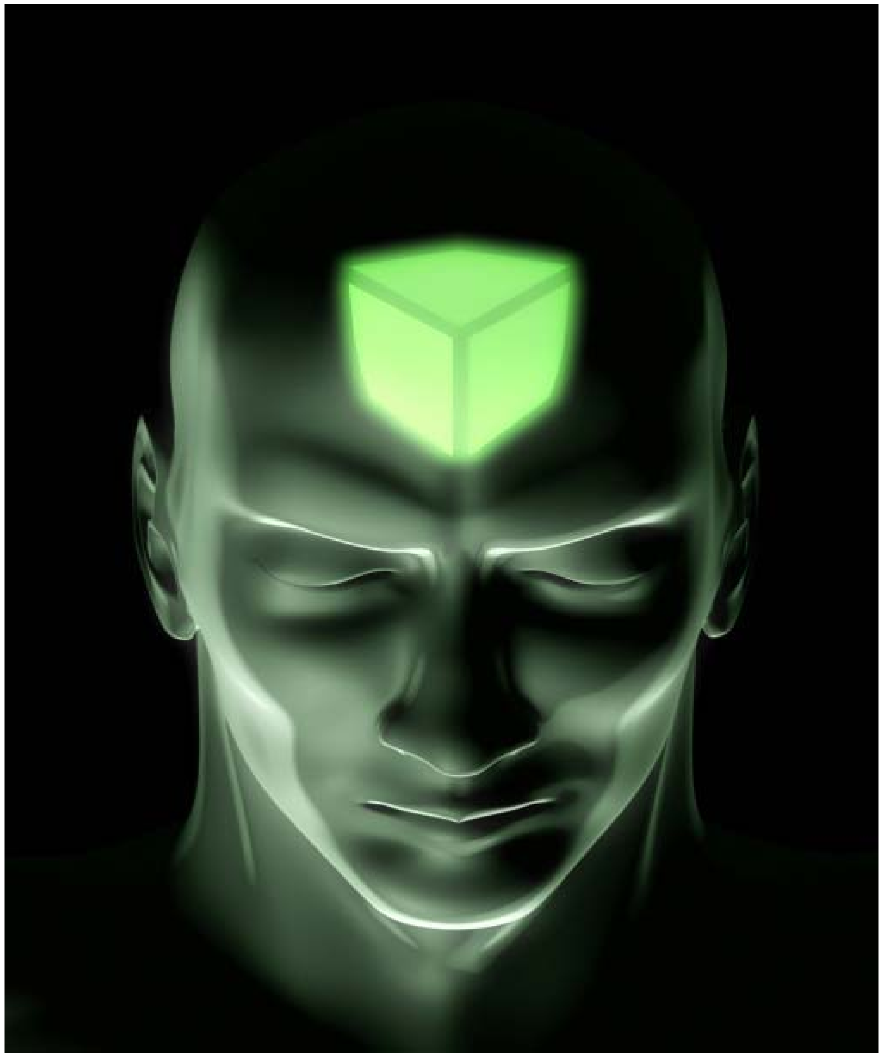
Schnecke -Thomas Kristof



Turtle - Marc Klooz

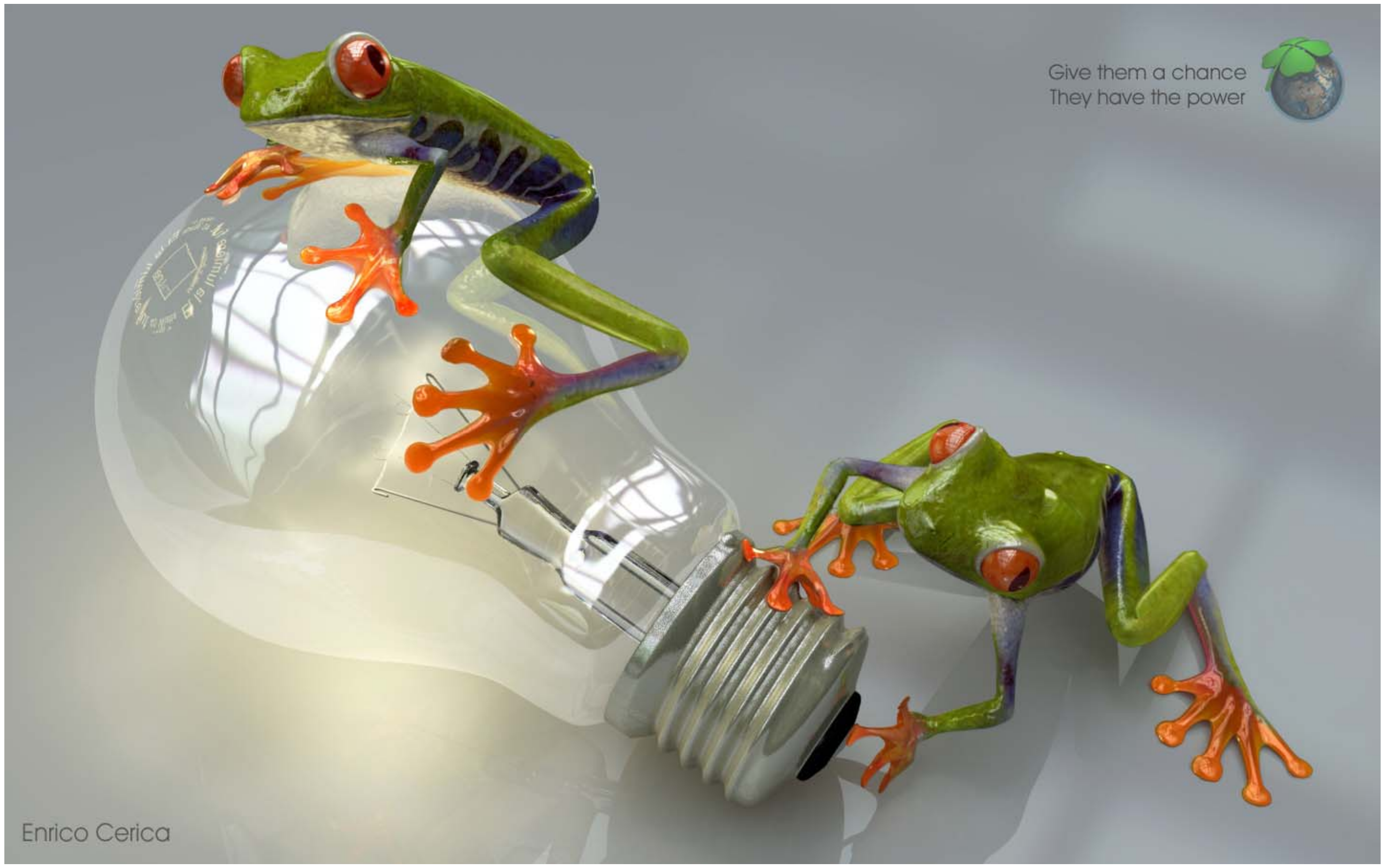


Klaus Kinski - Thorsten Schluter

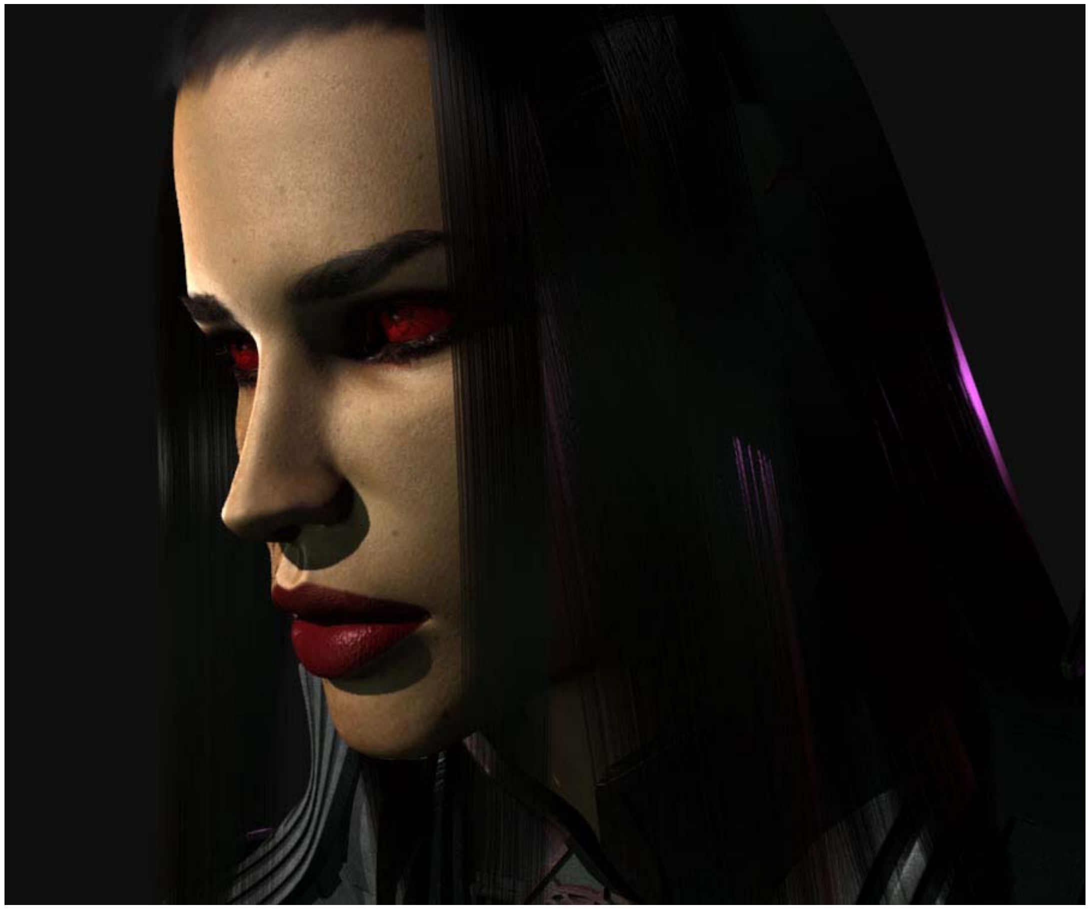


MC- Pedro Augusto

Monster - Pedro Augusto



They Have The Power -Enrico Cerica



face_hair_armor (Extinction Level Event) - Jeremy Ray

Issue#11 July 2007

Theme: Mechanical

Any thing mechanical, bad ass robo,
Heavy Duty Machinery.
Tanks

Disclaimer